The non-invasive localization of cardiac arrhythmias - a theoretical study

Stedman, Nicholas F.

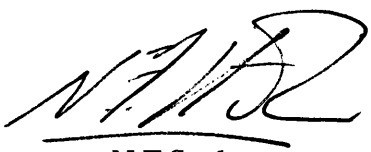
Award date:
1999

Awarding institution:
University of Bath

The Non-invasive Localization of Cardiac Arrythmias – a Theoretical Study

Submitted by Nicholas F. Stedman

for the degree of Doctor of Philosophy
of the University of Bath
July 30th, 1999



N.F.Stedman

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Preface

This thesis is dedicated to my wife, Susanne, for her enduring love and support, to my children, Caroline and Roger and my grandchildren, Hamish, Imogen and Kate. The jump from being a businessman and business strategy consultant to biophysical engineering research has been most interesting and rewarding.

I would like to thank my supervisor Professor Don Monro for inviting me, as a relatively mature student, to pursue these ideas and giving me the opportunity to meet many other researchers throughout the world in the same field.
I am particularly grateful to Dr Jukka Nenonen at the Dept. of Technical Physics at the Helsinki University of Technology who is a key member of the Biomedical Engineering team led by Professor Toivo Katila, for his help and for numerous visits to Helsinki enabling me to see at first hand the application of magnetocardiographic mapping at the Helsinki University Central Hospital.
My thanks also go to the team at Physikalisch-Technische Bundesanstalt (PTB) in Berlin led by Dr Lutz Trahms who invited me to present and discuss my ideas with them, as their own thinking is proceeding along similar lines.
Finally my thanks go to Professor Leonid Titomir, Head of the Laboratory of Biophysical Information at the Russian Academy of Sciences in Moscow whose timely book (Bioelectric and Biomagnetic Fields) jointly published in 1994 with his colleague Dr Peter Kneppo of the Slovak Academy of Sciences gave me a thorough introduction to the theory of this fascinating and vital area of research. The opportunity to visit his laboratory and see the medical research facilities in Moscow was much appreciated.

This PhD was funded by a full grant from the Engineering and Physical Sciences Research Council.

Summary

Since the first published human electrocardiogram in 1887, there have been major advances in cardiac diagnostics but, due to the overlapping characteristics of healthy and non-healthy subjects, prognostic capabilities are poor. Magnetocardiographic techniques have been added recently but have not changed the situation. Advances in minimally invasive cardiac surgical techniques have added importance to the localization, through techniques such as inverse modelling, of malign arrythmias, one of the major causes of sudden cardiac death.

Biolelectric sources are most commonly modelled mathematically using the dipole, the lowest order term of an infinite series but this representation has not been able to locate ischemic regions. Recent research on the electrophysiology of injury currents bordering these regions has shown the existence of weak circular currents. These currents can only be represented by higher mathematical terms and can give rise to magnetic signals with no accompanying electric signal. The hypothesis put forward is that it should be possible to locate such a source using only a quadrupole representation. This thesis shows that in theory localization is possible with good accuracy even when a much stronger nearby dipole source (possibly from healthy tissue) is present and also in the presence of a level of signal noise which would be expected to swamp the quadrupole influence. Localization accuracy falls the deeper the source.

New closed form equations are developed to solve the electric and magnetic fields due to a point quadrupole source and a little known technique to solve the inverse problem is applied. An insight is provided as to why the equations are capable of localizing a point quadrupole under relatively onerous conditions. The advantages gained from combining the electric and magnetic measurements are small when locating a quadrupole but there are different sensitivities.

This work must now be applied to real patient data.

INDEX

Symbols used in this Thesis:

Subscripts	
e	surface field measurement point
i	impressed
n,m,n' ,m'	summation indices
s or p	relating to source point
unscripted	any surface point
x,y,z r,θ,ψ r,ξ,φ	cartesian , spherical or spheroid co-ordinates ▪ when used with F = vector differentiation {with scale factors included}with respect to one or more co-ordinates in the order given by the subscripts. ▪ otherwise = a vector component
v	volume
∞	infinite medium value

Superscripts	
M	magnetic
α'	differential coefficient of α
T	a vector or matrix transpose
⊥	the orthogonal complement
-1	the matrix inverse
+	the pseudo inverse of the matrix

α,β	any co-ordinate
α	a vector of non-linear unknowns
χ^2	the least squares metric functional
ψ or β	any angle
δ_α^β	Kröneker delta function: $\delta_\alpha^\beta \begin{cases} =1 \ for \ \alpha=\beta \\ =0 \ for \ \alpha\neq\beta \end{cases}$
Φ	a matrix of position functions each of which is one or more expressions or infinite series
ε	the permittivity of free space
γ	source and field point co-ordinate expression
η	index dependent constant $\eta=\left(2-\delta_m^0\right)\frac{(n-m)!}{(n+m)!}$
λ	distance along image line
λ_{nm}	the elements of the harmonic expansion for V which relate to the source
μ_o	the magnetic permeability of free space
ϕ or V	scalar electric potential
ρ	distance from lead field point
σ	homogeneous medium conductivity
∇	gradient/vector differential operator (nabla)

a_j *(j=1..m)*	any multipole component
A_{nm}, B_{nm}	multipole components
$\mathcal{A}_{nm}$, $\mathcal{B}_{nm}$	multipole components for the prolate spheroid
$\overline{A}$	the magnetic vector potential
$\overline{B}$	the magnetic field
$\overline{D}$ D_x, D_y, D_z	the current dipole moment vector and its components
d	displacement between point sources or point dipoles
$D\left(f\left(\alpha^{(j)}\right)\right)$	the Fréchet derivative of $f\left(\alpha^{(j)}\right)$
D	the Hessian
E	Field point
$\overline{E}$	the electric field strength
$\overline{F}$	Lead field vector
F	the elements of λ_{nm} which are differentiable with respect to the source co-ordinates
h_α	a metric coefficient for the α plane
H_n or H_l	orthogonal component of the field gradient
I	current
$K_{(j)}$	a special matrix formed from the Fréchet derivative
$\hat{i}, \hat{j}, \hat{k}$	unit vectors in the x,y,z directions
$\overline{J}$	the total current density vector
$\overline{J}_i$ or $\overline{j}_i$	the impressed current density vector
$\overline{J}_v$	the volumetric current density vector caused by the electric field
$\tilde{O}$	the current octapole moment tensor
P_Φ	the orthogonal projector on the linear space spanned by the columns of Φ
$P_n^m(\,)$	Associated Legendre polynomial

$Q_n^m(\,)$	Associated Legendre polynomial of the second kind
$\tilde{Q}\ Q_{xx} .. Q_{zz}$	the current quadrupole moment tensor and its components
R_e or r_e	the line pointing from the source to the field point
R or r and $\bar{r}$	the line from the co-ordinate origin pointing to the field point and its vector representation
r_s and $\bar{r}_s$	the line from the co-ordinate origin pointing to the source point and its vector representation
r or rk	r is the instantaneous rank of a matrix, rk is the maximum or theoretical rank of the same matrix
R_p	a line pointing from one point on the surface of a sphere to another
s or S	the enclosing surface
S	a permutation matrix
$t_i\ (i=1..n)$	any one of *i* surface co-ordinates
T_{11} or T_{12}	a triangular or rectangular sub-matrix
$\hat{u}_\alpha$	any unit vector in the α plane
$\overline{u_\alpha}$	a unit vector in the co-ordinate direction α
v	a small volume, usually the heart region

1. Introduction

The non-invasive evaluation of the heart through the external detection of its electrical activity is now over 100 years old. Engineers have played a key role in the development of equipment and techniques to measure these electrical signals and over the years there have been many thousands of published papers from researchers of all nationalities. The prize of reducing or eliminating heart disease and its consequences is large indeed. The topic of this thesis is the detection of malign ventricular arrythmias, a sub-group of cardiac diseases, which are the most common cause of sudden cardiac death. From the statistics given by [Weismuller:1995] and [Moshage &Achenbach:1995] this category accounts for as many as one death every 5 minutes in the UK (20% of all deaths). Also it is not confined to the very old as, on figures published by the UK Government in July 1999, the total number of deaths from heart disease for under 75 year olds is 69,000 pa of which rhythm disorders are approximately 50%. Despite the enormous effort and close working with clinicians, electrocardiography (ECG) and its sister subject vectorcardiography (VCG) has never gained the status of being a stand alone predictive tool for various classes of heart disease. They are used instead in clinical diagnosis to provide support or confirmation. The reasons for this are well known amongst the medical fraternity and were outlined by [McFee and Baule:1972] who said that the range of normal ECGs was so wide that applied diagnostic criteria gave too many false positives and that the very concept of 'normal' could be an illusion.

This has not dissuaded the many electrocardiologists who continue to hunt for ECG patterns and statistical relationships between these patterns and the various coronary diseases. My own view is that, despite the huge effort and long history, the subject is still at a primitive stage. The electophysiology of injury sites in the heart is not well understood and the instrumentation, detection and analysis of the complex cardiac signal is still developing. The current rather blunt approach to the localization of a particular malfunction seems to be similar to asking a tone deaf person to pick out the one instrument that is off-key in a full orchestral concert. The signals from the heart are a compound set of signals usually generated from a large area of activity, mostly performing normally (see Fig 1.1):

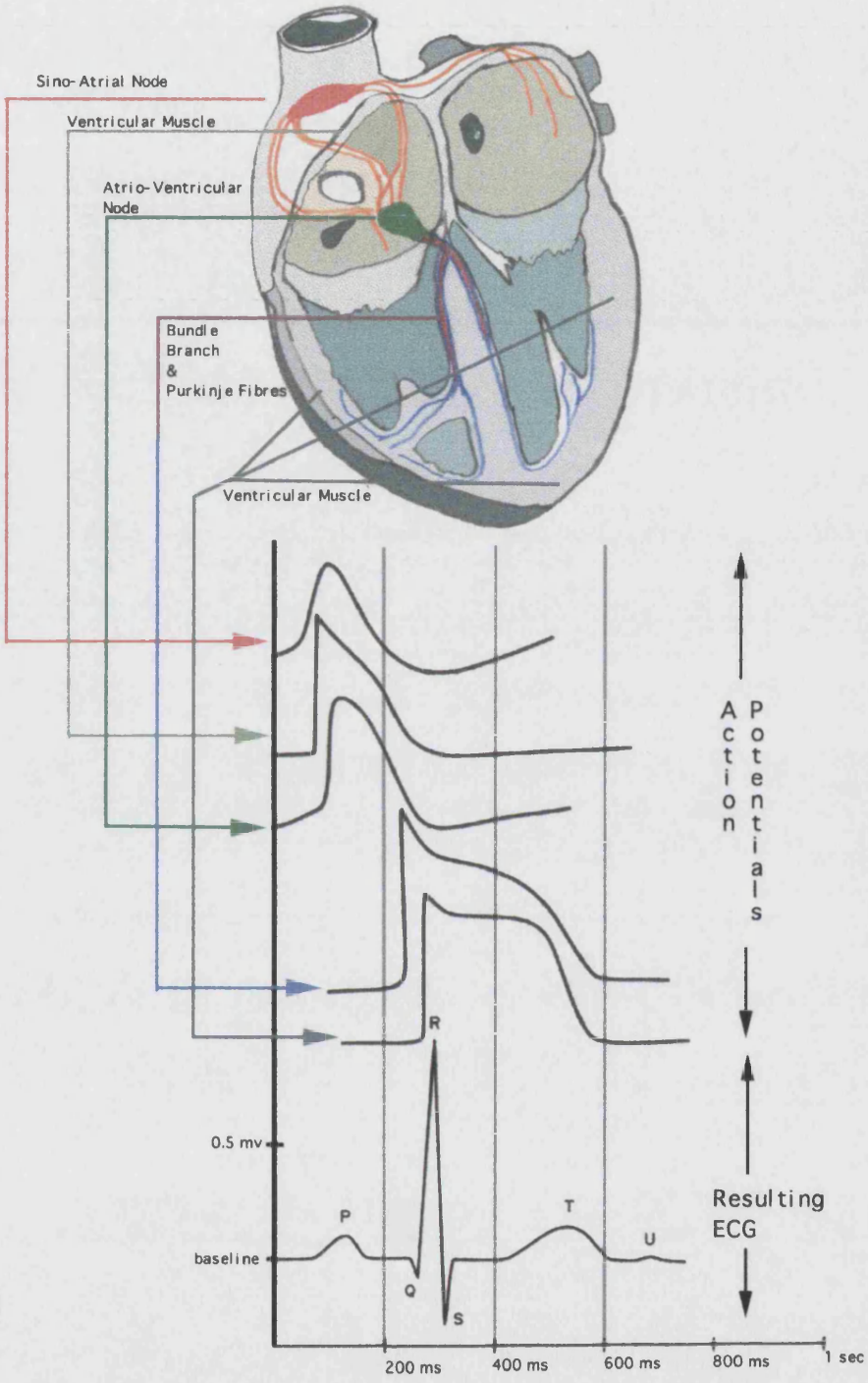


Fig 1.1 The normal heart and its electrical activity

Abnormal, and probably quite weak, signals are added and subtracted by different cardiac pathologies at different points in the cardiac cycle. The signals are 'muffled' as they pass through complex layers of different (and anisotropic) conductivity before being detected at a finite number of points by instruments which add white noise. Is it not surprising that anything coherent at all is detected?

There is no reason to believe that bioelectricity, which is a sequence of ionic currents, does not follow the normal laws of physics, it is just that the human body is electrically very complex. However, advances in instrumentation, in computer modelling and in electrophysiology, which are touched on below, will lead to a better understanding of the nature of the bioelectric behaviour of diseased cardiac tissue and to a better analysis of the entire cardiac signal.

This introduction will give a brief history, as is traditional, of the subject of inverse ECG and the more recent MCG (magnetocardiographic) modelling. It will go on to describe different approaches to the subject of inverse modelling and to recent developments in electrophysiology which may have some significance. The purpose and outline of this study will then be described.

1.1 Background

The representation of the electrical behaviour of cardiac cells by sets of current sources and sinks has been accepted as the only realistic way of handling the mathematics of electromagnetic theory. By 1954 it was already well known that it was not possible to work out a unique pattern of the electrical sources and sinks of current in a homogeneous bounded volume conductor from the knowledge only of the body surface potentials. A single set of surface potentials could arise from an infinite number of arrangements of sources and sinks. The heart was instead represented by a single dipole vector at a fixed location and, from a few ECG measurements, its vector magnitude and direction was estimated at different time intervals. [Gabor and Nelson:1954] showed that it was possible to reconstruct this dipole and, under certain assumptions, its location from the integration of the potentials over the bounding surface. [Geselowitz:1960] found that a collection of multipoles (a multipole expansion) at a single point inside a bounded homogeneous conductor can uniquely represent a distributed surface potential and can be called an equivalent generator. The dipole (lowest) term of a multipole expansion does

not contain any location information but the quadrupole and higher terms do. By finding the position which minimised these higher terms, the dipole can be located. He also found that if these multipoles are confined to the smallest possible region inside the volume conductor, the multipole expansion will converge (ie successively higher terms will contribute decreasing amounts of potential) at every point outside this equivalent source region. If a location is found and there are still contributions from higher terms then this is a measure of the non-dipolarity of the equivalent source. He hints that small amounts of higher terms may have to be included depending on the outcome of diagnostic studies but, if many terms were needed, the computational burden would probably rule out this approach as having any clinical diagnostic future.

The ideas were refined with two quadrupole theorems [Geselowitz:1965]. Moving a point multipole expansion to a new location changes all the terms of the expansion. A translational move of a quadrupole not only changes the quadrupole components but adds (or subtracts from) a dipole term. A rotation of a quadrupole alters only the diagonal components of the quadrupole and does not add further terms. As mentioned above, the best fit dipole is found by minimising the quadrupole term which automatically implies that higher terms are minimised. He states that there is some evidence that people with heart disease have a smaller dipole vector. He also points out that the best fit location for a dipole is not going to be the same as the best fit location for a dipole and a quadrupole. It depends which dominates at what time interval, but they should be close.

Starting from the foundation theory of Helmholtz, a good summary of the basic assumptions made in arriving at the general equations for solving the bioelectric inverse is given by [Geselowitz:1967]. This paper points out that one of the key assumptions of cardiac work is that of quasi-static potential distribution, or tissue capacitance can be neglected, implying that charges on boundaries instantly redistribute themselves to reflect changes in the underlying source. Using Green's Theorem for the expansion of the space vector, he develops general equations for the case of an inhomogeneous conductor where conductivity discontinuities can be accounted for by a surface integration, around the discontinuity, of the potential that would exist at that boundary if the sources were in an infinite homogeneous volume conductor. Each conductivity boundary is effectively represented in the infinite medium by the addition of a dipole layer.

Multipoles obey superposition as pointed by [Brody:1968]. Therefore sets of dipoles may be taken to represent an equivalent cardiac generator. He was able to obtain an exact fit of the potential due to a dipole and a quadrupole by taking a dipole pair and carrying out a least squares minimisation through translations and rotations of the pair about their mid point.

According to [Arthur and Geslowitz:1970] the dipole and quadrupole terms contain most of the information about the source location. They showed that discontinuities due to blood mass and the lungs caused the location to move closer to the centre of the heart and this effect increased the shallower the source. Maximum displacements from true of 1cm were noted. They confirmed what had come to be known as the Brody Effect: namely, that the effect of low resistance blood mass causes the radial dipole component to be emphasised while the tangential component is diminished.

In the 1960s the first detection of the magnetic field had been reported by [Baule and McFee:1963] who later on developed a magnetic lead field theory analogous to electrical lead fields [Baule and McFee:1965]. This magnetic field is extremely weak – about one millionth of the earth's magnetic field. By the early 1970's the first magnetic measurements had been made using a highly sensitive supercooled Josephson junction today known as the SQUID (Superconducting Quantum Interference Device). The promise held out by MCG and pointed out by [McFee and Baule:1972] in a comprehensive review, was that new information about the source might become available. In particular, circular arrangements of dipoles would be detectable by MCG but not by ECG. It was also noted that the Brody effect is not apparent in MCG work but rather the reverse, that tangential dipoles will be emphasised. The anisotropy of heart muscle tends to emphasise this anti-Brody effect. This greater sensitivity to transverse currents might be more useful in detecting abnormalities as EMFs in normal subjects tend to be radial. They also commented on the likely effect on the ECG of an infarct. This places an electrical hole in the myocardium and, according to cup shaped (effectively the active shape of the myocardium) double layer theory, this should produce 2 dipoles – one at the cup mouth (the top of the heart) and the other at the site of the infarct, both pointing inwards. They also predicted that potentials produced by injury

currents are likely to be small and undetectable by ECG but more likely detectable by MCG.

The fundamental difference between ECG and MCG was pointed out by [Plonsey:1972] in that, with ECG, the multipole expansion of the source's divergence is used to recreate surface potential measurements while, with MCG, the multipole expansion of the source's curl is used. The conductivity of the volume conductor does not take part in the magnetic field equation. However, conductivity boundaries produce a secondary dipolar current which does contribute to the magnetic field. He points out that according to Helmholtz's theorem a vector field, such as the current source in the heart, is completely specified by its divergence and its curl. The implication is that ECGs and MCGs are complementary and are both needed to provide a complete description of the source. This debate continues today.

A study by [Geselowitz and Miller:1973] was based on the electric and magnetic fields created by 20 dipoles in a homogeneous sphere, at fixed positions mostly near the centre. They attempted to recreate the dipole magnitudes from combined inverse modelling. They added white noise to the field data at 1% and 5% (which would give 40dB and 30dB SNR respectively). They concluded that the solution drastically deteriorated with the addition of small amounts of noise but noted that accuracy improved for combined ECG and MCG data as against separate inverses.

A theoretical paper by [Grynszpan and Geselowitz:1973] developed a set of general expressions for the electric and magnetic multipole expansion using spherical harmonic functions. More interestingly, as so little attention had been given to it, based on earlier work by [Yeh and Martinek:1957], they produced an expression for the magnetic dipole moment for a current dipole in a conducting prolate spheroid. The interest in spheroids and a spheroidal harmonic expansion is due to the fact that the torso (and also the head, since this area of cardiac and neurological studies have closely associated theory) is not a sphere but closer to a spheroid. Even better would be an ellipsoid but there is almost no published work for this geometry in this field but see [Blimke et al:1997].

The effect of both geometry and inhomogeneities on the ECG was studied by [Rudy and Plonsey:1979] in a concentric spheres model. The source was a double layer spherical cap as part of a complete eccentric sphere within a larger sphere. The internal eccentric

sphere represented heart muscle and intracavitary blood, the remaining space inside the larger sphere represented lungs while the outer layer represented a muscle and fat layer. This complicated arrangement, although still far from the real world, was analytically solvable. They found that, compared to a homogeneous solution, the general pattern of the potential was broadly the same with the characteristic dipole two peak surface map. They concurred with other researchers that the appearance of other peaks is almost certainly due to non-dipolar sources and not due to inhomogeneities. They confirmed the Brody Effect (due to the blood mass) but also found that the pericardium and outer body muscle layer to an extent counteracted this, reducing the importance of this phenomenon. The muscle layer in their view, however, should be included in ECG modelling.

In an overview of MCG, [Geselowitz:1979] had still not concluded that this expensive technique produced any new information over ECG. How to handle the vector nature of the field was still unresolved. Only in the area of injury DC type currents (detectable by a SQUID) was there any promise. Signal noise was still a problem.

Attention now focused on inverse models and realistic torso modelling, and almost all subsequent papers used either a simple electrical dipole or a magnetic dipole as an equivalent generator. At this stage of this thesis it is noted that the magnetic dipole is a part of the current quadrupole but undetectable by ECG.

1.2 Approaches to Inverse Modelling

Inverse modelling is based on the construction of a forward model whereby a mathematical relationship is created between an assumed equivalent current source in the cardiac region and the surface of the body, allowing the electric potentials and/or the magnetic field to be computed. The creation of this mathematical model is based on Maxwell's equations describing the propagation of electromagnetic waves. The geometric relationship between the current source location in the heart region and the body surface measurement points is a key part of the process. More recently attention has been focused on the nature of the layers of intervening tissue and the effect of the irregular boundaries between layers with very different electrical properties and incorporating this into the forward equations. Nonetheless, despite the complexities, a given set of assumptions concerning the source and surrounding tissues will always lead to a unique surface profile of the electrical potential and magnetic field. It can be shown mathematically that the reverse is not true – namely, a given profile on the surface can unfortunately be created by an infinite number of equivalent current source arrangements. The only way out of this is to define the source in such a way that the mathematical degrees of freedom are sufficiently restricted to allow a unique estimate to be made of a limited number of its parameters. Those parameters must be chosen to yield the required diagnostic information. For example, is an arrythmogenic site or an accessory pathway being located, is a characteristic QRS profile being matched, or many other possible questions. A large number of papers have ignored this important point.

With a few exceptions, most papers on the subject have assumed the simplest possible source, the point dipole, which consists of a point current source and an almost coincident point current sink. The reason for this is that in healthy heart tissue this closely follows the cellular activation process which approximately spreads out from the endo- to the epicardial surface in a chain like sequence of activated dipoles. The modelling of this wave and its consequent surface map, particularly in relation to various pathologies, is the target of many researchers. However, even at the early stages of the cardiac cycle a very large number of cells are active as pointed out by [Killmann et al:1995] who estimated from a theoretical study that 12ms into the cycle of a Wolff-Parkinson-White

(WPW) syndrome patient (where an abnormal conduction path exists between the atrium and ventricles) an area approximately 0.7 cm^2 containing approximately 70,000 cells was active. As long as the wavefront maintains a linear progression then a dipole may be a reasonable representation. Any pathology which disrupts the sequence may be better modelled in some other way.

The forward models fall into two broad categories as dictated by the mathematical solutions to the modified and simplified Maxwell's equations : volume models and surface models.

Volume models create a 3 dimensional model of the torso out of discrete blocks and assign common electrical properties to groups of blocks. Each block is related to its neighbours through a simplification of the Maxwell differential equations. The most popular method is the finite element model which can either relate the edges of a block to its neighbours or the nodes of a block. The latter is almost universally used in bioelectric work where boundaries are irregular. The finite difference approach defines blocks but interconnects them with a 3 dimensional wireframe of electrical resistors. A third method is the finite volume estimate which is close to the finite element method but avoids one of the *a priori* assumptions in the finite element method. Both require very large matrices to be solved and are computationally intensive. A good but brief description of the two methods is given by [Gulrajani:1997] and a good example of the finite difference approach can be found in [Budgett:1995]. The big advantage of the volume models is that directional or anisotropic conductivity properties of the internal tissues can be easily modelled. Also one of the mathematical problems, matrix singularities, can be avoided by the use of Dirichlet boundary conditions. The volume solution is very useful for modelling the source region where fine discretizations can be used and where an understanding of the complex anisotropies is vital. According to [Paulsen:1997] further advances on mesh generation are needed so that fine and coarse meshes can be combined and avoid the tendency to spurious solutions as element numbers are increased.

Surface models use the integral equation solutions to Maxwell's equations. Only the surfaces of conductivity boundaries are discretized usually as interlocking triangles. The potential estimates are made at the centre of the triangle, or more often at the vertex of each triangle as there are approximately half the number of vertices as there are triangles

for most surfaces. Anisotropic regions are converted to isotropic equivalent approximations. There are fewer matrix elements than the volume models but every element is related to every other element which leads to a full solution-matrix. The equations are solved through the use of a Neumann boundary condition which unfortunately leads to a matrix singularity. A deflation technique, whereby the average surface potential can be zeroed, is used to side-step the problem.

The integral equations for the surface field are derived by the application of Green's second identity which is the vector relationship of two points in space defined by any coordinate system. The usual expansion of this space vector is an infinite series spherical expansion. The potential at any point in space is related to this vector through the medium conductivity and the intensity of the source. The equations can be solved analytically but only for simple geometric shapes such as a cylinder, sphere, spheroid or ellipsoid (and the so called half-space). Even then, the solutions are cumbersome except for the lowest order term, the dipole term. It is the current dipole with the spherical expansion which is almost universally used for the electric potential solution. The magnetic dipole consists of opposing pairs of current dipoles and is part of the full quadrupole term, the second term of the expansion. This is the most commonly applied solution for magnetic fields on a surface. The solutions for a sphere can be found in [Gryszpan:1971] and [Cuffin and Cohen:1977]. Analytical solutions can incorporate regions of differing conductivity but only isotropic homogeneous regions of very simple geometry (eg see *op cit* [Rudy and Plonsey:1979]).

The point current dipole has not proved very useful for the localization of a single event. [McLeod et al:1995] thought it might yet be possible but commented "... discrete sites of epicardial activation merge into single features of body surface maps and the relative strength of posterior cardiac events is lost ..." . The single dipole approach has been most successful in the case of pathologies where a specific region of activity can be isolated such as a bundle branch block or WPW syndrome where the abnormal conduction path is being sought [Nenonen et al:1993].

Once the dipole wavefront is spread out in the wall of the myocardium, point dipole representations are of less use. Although the idea is not new, distributed source models have only come to the fore in the last 10 years due to the intense computational load.

Multiple dipole solutions have been attempted and are more applicable in brain studies where electrical activity is thought to be almost exclusively dipolar: see, for example, [Kavenagh et al:1978]. In the cardiac field, the multiple dipole approach has been taken to 352 unknown dipoles in an attempt to map cardiac surface potentials [Ferguson and Stroink:1995]. The uniform double layer (UDL) is possibly a more interesting approach [Huiskamp:1988] where the dipolar wave is used as the source model on the basis that depolarization is largely a 2 dimensional surface travelling in a normal direction to the cardiac surface from the endocardium. The dipole density is uniform across this moving surface and the time intervals when the wave breaks through the epicardial surface, generating a potential at that point and potentials across the body surface, is computed inversely from those body surface potentials. An isochrone map of the epicardial pattern can be drawn. Until recently, [Oostendorp et al:1997] validation of the results of the UDL source model with invasive data has been lacking. A further criticism of this method [Clerc:1976] is that for UDL to work, the transverse and along fibre conductivities would have to be broadly the same, which he claimed that they are not. Myocardial muscle is known to be anisotropic but there is conflicting evidence as to how large the differences are in relation to propagation velocities. Recent work [Baynham and Knisley:1997] confirmed that transverse to along fibre resistivity is of the order of 2:1.

Some of the above points on UDL are taken from [van Oosterom:1996] but as a summary of the state of the art of modelling in general it is worth quoting a small part of the introduction to his paper:

> "Apart from satisfying scientific curiosity, the objective of modelling the cardiac field must be the improvement of clinical diagnostic procedures. Although this may seem obvious, the direction the modelling world is taking seems to be diverging from this objective. In spite of the major progress of insight obtained through modelling, surprisingly little of this seems to have had an impact on clinical electrocardiology"

1.3 Recent Developments

Arrythmias fall into two broad groups: pacemaker activity when a group of cells begins to depolarize automatically, and re-entrant excitation when the parts of the wavefront on

meeting ischemic regions can move around and back on themselves re-exciting cells that have just repolarized [Weiss:1997]. It is the latter group which is of most interest and the most common mechanism of cardiac arrythmias. Conventional body surface ECG is insufficiently sensitive for defining clear criteria for the diagnosis of infarctions and ischemias. [Startt/Selvester et al:1989] claim that between 40% to 50% of all confluent infarcts 2cm or larger are not associated with classic QRS changes due to infarction (see Fig. 1.2). As ischemias develop there is a clear progression in epicardial potential characteristics, which of course are not easily measured:

a) decrease in the magnitude of the resting potential
b) decrease in the amplitude of the action potential (AP)
c) shortening or prolongation of the AP
d) increase in the AP rise time

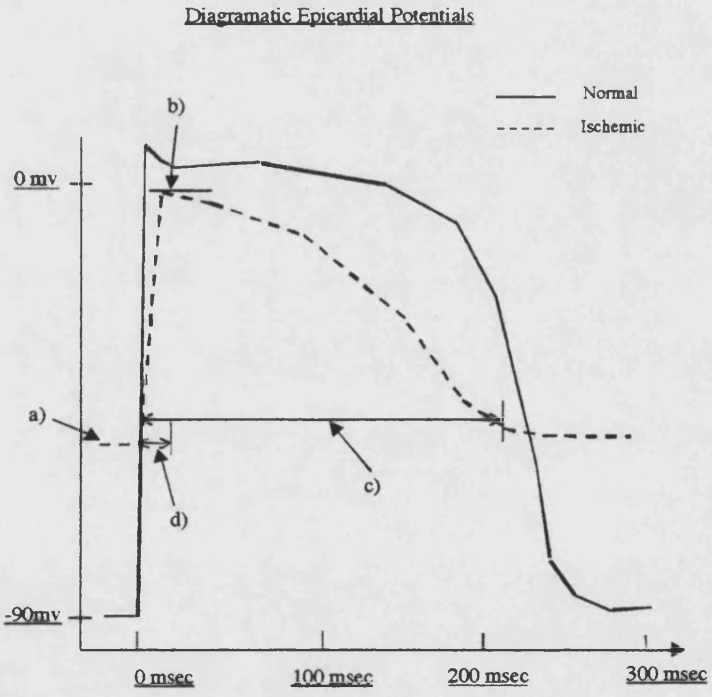
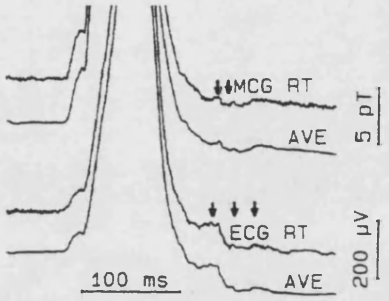


Fig 1.2 The action potential of the ischemic heart

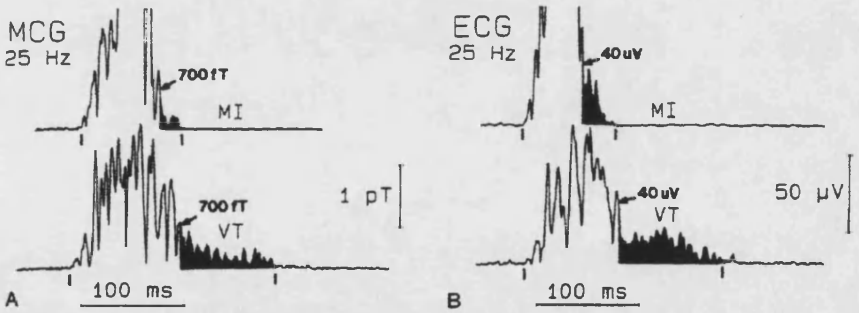
Some of the re-entrant mechanisms of ventricular arrythmias have been known about for 80 years after work on animals showed anatomical abnormalities in the myocardium,

where excitable fibre bundles had formed into rings. By 1985 it was known that re-entry plays a major role in the genesis of most ventricular arrythmias [Gomes et al:1985]. The characteristic ECG signal of such arrythmias is a low amplitude high frequency (above 80Hz) signal at the end of the QRS phase leading to a lengthening of this complex. This fractionation of the electrogram had already been associated with conditions that favour re-entry: slow conduction, unidirectional conduction block and recovery of excitability. Amplitudes during fractionation were <20μv . A similar pattern emerges with MCG [Mäkijärvi et al:1993] with amplitudes <700fT (see Figs. 1.3 and 1.4):



Real time and averaged unfiltered QRS complexes in the high-resolution MCG and high-resolution ECG of a patient with VT. Abnormal, low-amplitude late ventricular activity seen in each recording is indicated by arrows. AVE = averaged; ECG = electrocardiogram; MCG = magneto-cardiogram; RT = real time.
By kind permission of the authors [Mäkijärvi et al:1993]

Fig. 1.3 ECG and MCG Fractionation – unfiltered



(A) Filtered (25-Hz high-pass) QRS complex of the high-resolution MCG recordings of a patient with MI and a patient with VT. The QRS duration is 97 ms for the MI patient and 189 ms for the VT patient. The root-mean-square-60 values are 2,320 fT and 130 fT, respectively. The onset and offset of the QRS are indicated by vertical bars. The shaded area at the end of the QRS complex shows the late ventricular activity (<700 fT). MCG = magnetocardiogram; MI = myocardial infarction; VT = ventricular tachycardia. (B) Filtered (25-Hz high-pass) QRS complex of the high-resolution ECG recordings of the same patient with MI and the same patient with VT as in A. The QRS duration is 80 ms for the MI patient and 203 ms for the VT patient. The root-mean-square-40 values are 133 μV and 5.7 μV, respectively. The QRS onset and offset are indicated by vertical bars. The shaded area at the end of the QRS complex shows the late ventricular activity (<40 μV). ECG = electrocardiogram; MI = myocardial infarction; VT = ventricular tachycardia.
By kind permission of the authors [Mäkijärvi et al:1993]

Fig. 1.4 ECG and MCG Fractionation – filtered

Since these discoveries research has been carried out in trying to understand the electrophysiological mechanisms at work and the size and structure of re-entrant loops. Work on the atrial epicardial surface during atrial flutter in a canine model [Schoels:1990] showed mostly single loops but also some figure eight re-entry. Arcs of circulation around central blocks were in the range 25 – 35mm although it was noted that electrodes as close as 2mm displayed activation time differences of 50ms and that re-entrant circuits of <5-8mm could have been missed. The electrophysiological properties of re-entrant pathways are not understood [Schoels et al:1991] and are non-uniform functional changes in the underlying pathology. [Saumarez et al:1995] thought that discrete conduction pathways existed due to variations in fibre diameter leading to altered conduction velocities and variations in refractoriness plus anistropies caused by fibrosis. The mechanism forming these circulatory or vortex currents seems to involve deviated, reflected and slowed down wave paths at infarct boundaries which curve round to meet cells nearing the end of their refractory phase and therefore re-excitable. Vortex currents have also been reported during repolarization [Trahms et al:1996] detectable only by MCG which would indicate paired anti-symmetric quadrupolar behaviour.

The possibility of spatial location of these low amplitude magnetic fields was pointed out by [Weismuller et al:1995]. A more detailed study [Muller et al:1999] used a fragmentation scoring system in conjunction with 2 dimensional MCG mapping. Patients with ventricular tachycardias and ventricular fibrillation had characteristic fragmentation maps with peak fragmentation regions displaced from those of normal subjects. They also thought that shorter abnormal activations may cause micro re-entry which remains hidden in the QRS complex.

Modelling of the mechanisms at work has been attempted by [Sanfelici:1998] who modelled the membrane ionic currents in general anisotropic conditions. She demonstrated the anomalous behaviour of the excitation wavefront in the region of an infarct. The shape of the wavefront could generate fractionated electrograms with multiple deflections. [Lin et al:1999] also modelled a quatrefoil re-entry pattern which they had photographed using fluorescent dyes in a rabbit's heart. Quatrefoil patterns are four symmetrical sets of opposing circulation covering an area not greater than 20mm by

10 mm. They were able to reproduce the pattern in their model using various arrangements of functional block and unequal anisotropies.

1.4 Objectives of this Thesis

From the above work it would seem that source models which include quadrupolar current sources will be able to account for such re-entrant activity and might be better at locating the sites of injury currents . Re-entrant loops are small and can last for more than one cycle [Lin et al:1999] which may provide the right signal conditions where pathologies are at an early stage and can be localized. Where micro re-entry is widespread, such as in patients who have suffered ventricular fibrillation arrest [Saumarez et al:1995], localization may not prove possible.

At this introductory stage it must be emphasised that a quadrupole is not a physical entity, it is a mathematical construct (so also is the dipole). Indeed, it is only one term from an infinite series expansion. Visualisation of the dipole is easy and has some analogy at cellular level. Visualisation of a quadrupole source is less easy but [Wikswo and Swinney:1984] have presented some drawings which are based on the different derivations of multipoles. Although the dipole and quadrupole will be discussed as if real, the notion of a stand alone quadrupole has no physical meaning, any more than a stand alone dipole or even the two together. The introduction of conductivity boundaries complicates the issue by introducing equivalent point sources which are coincident with the primary source and completely fictitious, but nonetheless useful as a mathematical representation of the field generators. This should be kept in mind when reading this thesis, or any treatise on the subject. Dipoles and other multipoles seem to appear and disappear depending on the how the mathematics are constructed and how well real world data fits the mathematical model.

Examining the data so far published, the quadrupole component strengths are likely to be weak in relation to average or peak QRS activity, possibly in the range of minus 20dB to 40dB [Mäkijärvi et al:1993]. Also, if the normal activation wavefront is still active, an electrical centre for the dipole could be present. The spatial relationship of the two equivalent (fictitious) sources is unknown. Signal noise is still a problem in MCG signals although improving steadily and may one day become better than ECG signal noise which is limited by patient noise.

The questions that need to be resolved for the localization of cardiac arrythmias are:

- is it possible to detect a weak point quadrupole current source in the presence of noise and a nearby stronger point dipole source?
- what is the impact of different noise levels on the location of a quadrupole?
- inverse localization models are known to be sensitive to depth, so at what depth can the source be located assuming a realistic level of signal noise?
- can localization accuracy be improved with combined MCG and ECG data?

As described above, there are elements of these questions partially addressed in other studies (a fuller review is presented in the conclusion to Section 6) but many studies are inconclusive on what are the causes of localization error. However, it appears that no full theoretical study has been carried out which answers them in a way that is unambiguous. Indeed very little attention has been paid to higher order sources: [Fiesler:1995] "As far as can be ascertained, this is the first report of the forward solution for the magnetic field of multipole contributions higher than the dipole, using the homogeneous conducting sphere". Sadly, the report only covered a forward calculation for the radial component of the magnetic field and did not cover the full analysis of the boundary effect.

This thesis will be based on full analytical electric and magnetic solutions to the forward problem so that 'exact' data can be generated for use in the inverse model. The word 'exact' must be treated with caution as numerous assumptions and mathematical truncations may have to be made in even reaching a workable analytical solution. The inverse model must be run in such a way that it as close as possible to real life. Unfortunately, such a model may not be usable in real patient analysis as it is generally agreed that regions of different conductivity will have to be modelled accurately before inverse models yield accurate and reliable diagnostic information. Does this view apply to quadrupole localization?

Section 2 will outline multipole theory and Section 3 develops the forward spherical equations which will be used for the surface ECG and MCG fields from separate or coincident arbitrarily located dipole and quadrupole sources. A similar set of equations is developed in Section 4 for the prolate spheroid using a spheroidal expansion of Green's function. Only the spherical solutions are evaluated in this thesis. Section 5 describes a powerful method of separating the linear from non-linear unknown variables in an

iterative inverse solution. The results from computer modelling the equations using this inverse method are presented in Section 6 followed by the somewhat unexpected conclusions and suggested future work.

2. Review of Multipole Theory

Bioelectric problems have the distinguishing feature that all electrical (ionic) sources are confined to a small volume conductor within a larger insulated volume conductor and distributed throughout smaller volumes within that conductor. Electrical engineering problems usually have identifiable external electromotive forces.

The general theory governing this situation will only be described briefly. There are a number of classical textbooks covering the theory: [Stratton] [Morse and Feshbach:1953]. A good description of the theory as applied to the bioelectric problem can be found in [Plonsey:1989].

The volumetric form of Ohm's law in a homogeneous conductor where an electric field is present is:

$$\bar{J}_v = \sigma \bar{E} \qquad 2.1$$

where : $\bar{J}_v$ is the ohmic current density Am^{-2}

$\bar{E}$ is the electric field Vm^{-1}

σ is the conductivity Sm^{-1}

To this can be added the impressed current $\bar{J}_i$ from the source region (the heart) to give the total current $\bar{J}$. In any closed region throughout the volume the net inflow and outflow of current cancels so that:

$$\nabla.\left(\bar{J}_i + \sigma \bar{E}\right) = 0 \qquad 2.2$$

From Maxwell's equations the negative gradient of the scalar potential (ϕ) is proportional to the electric field plus the time varying gradient of the vector potential. However, the latter term in the relatively low frequency bioelectric generator regime is negligible and is omitted, so that:

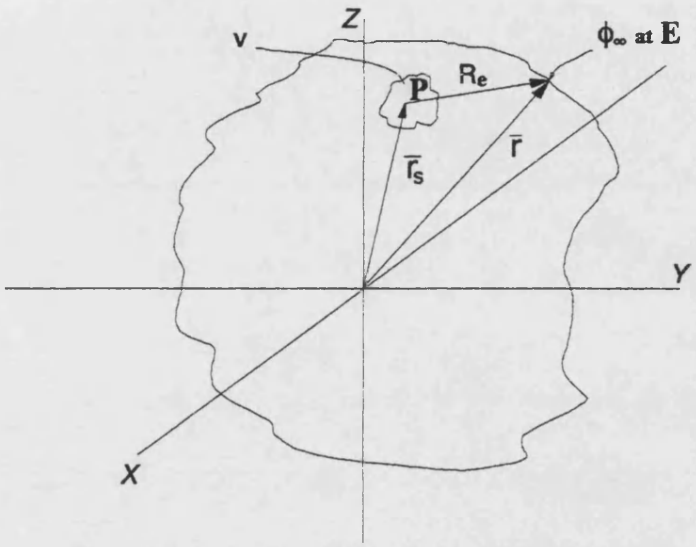
$$\bar{E} = -\nabla\phi \qquad 2.3$$

Substituting 2.3 in 2.2 gives Poisson's equation:

$$\nabla^2\phi = \left(\nabla.\bar{J}_i\right)/\sigma \qquad 2.4$$

and the integral solution to this gives the potential at any point in the infinite homogeneous medium:

$$\phi_\infty = \frac{1}{4\pi\sigma}\int_v -\frac{\nabla.\bar{J}_i}{R_e}\ dv \qquad 2.5$$



R_e is the distance from any point P in the source volume v to any field point E on the outer surface. From an arbitrary origin:

$$\frac{1}{R_e} = \frac{1}{|\bar{r} - \bar{r}_s|} \qquad 2.6$$

Using the vector identity:

$$\nabla.(a\bar{\alpha}) = a\nabla.\bar{\alpha} + \bar{\alpha}.(\nabla a) \qquad 2.7$$

and Gauss' divergence theorem:

$$\int_v \nabla.\bar{\alpha}\ dv = \oint_s \bar{\alpha}.d\bar{s} \qquad 2.8$$

then, with $\bar{\alpha} = \bar{J}_i$ and $a = \frac{1}{R_e}$, 2.5 becomes:

$$\phi_\infty = \frac{1}{4\pi\sigma}\int_v \bar{J}_i.\nabla_p\left(\frac{1}{R_e}\right)\ dv \qquad 2.9$$

since the closed surface integral $\oint_s \frac{\bar{j}_i}{R_e} . d\bar{s}$ of the source current is zero. The subscript p means that the gradient operator acts at point P in the source.

The simplest representation of the source is the dipole, which is a point source and a point sink infinitely close with infinite strength, giving a finite dipole (or first) moment about its centre. The first moment is a vector $\bar{D}$.

The gradient of the space vector can be expanded:

$$\frac{1}{R_e} = \frac{1}{|\bar{r} - \bar{r}_s|} = \frac{1}{r} + \frac{\hat{r} . \bar{r}_s}{r^2} + \frac{3(\hat{r} . \bar{r}_s)^2 - r_s^2}{2r^3} + \ldots\ldots \qquad 2.10$$

where $\hat{r} = \frac{\bar{r}}{r}$ is the unit vector in the r direction

and $r = |\bar{r}| \quad r_s = |\bar{r}_s|$

A first approximation of the potential due to the unknown arbitrarily located dipole source can now be made:

$$\phi_\infty = \frac{1}{4\pi\sigma} \bar{D} . \nabla \left(\frac{1}{R_e} \right) \qquad 2.11$$

This is normally written as a set of multipole components in ascending order. The first two orders are presented here and the third order can be seen in, for example, [Titomir and Kneppo:1994]:

$$\phi_1 = \frac{1}{4\pi\sigma r^3} (A_{10} z + A_{11} x + B_{11} y)$$

$$\phi_2 = \frac{1}{4\pi\sigma r^5} \begin{bmatrix} -\frac{1}{2} A_{20} (x^2 + y^2 - 2z^2) \\ +3A_{21} xz + 3B_{21} yz \\ +3A_{22} (x^2 - y^2) + 6B_{22} xy \end{bmatrix} \qquad 2.12$$

Here (x,y,z) is the field point E.

The multipole components which represent the dipole source are:

$$A_{10} = D_z \quad A_{11} = D_x \quad B_{11} = D_y$$

$$A_{20} = -D_x x_s - D_y y_s + 2D_z z_s$$
$$A_{21} = D_x z_s + D_z x_s \qquad B_{21} = D_y z_s + D_z y_s \qquad 2.13$$
$$A_{22} = \frac{1}{2}\left(D_x x_s + D_y y_s\right) \quad B_{22} = \frac{1}{2}\left(D_x y_s + D_y x_s\right)$$

where(x_s, y_s, z_s) are the source coordinates at point P.

The multipole source components are more concisely expressed in spherical coordinates using the spherical multpole expansion:

$$\begin{Bmatrix} A_{nm} \\ B_{nm} \end{Bmatrix} = \bar{D} \cdot \nabla_p \left\{ \sum_{n=1}^{\infty} \sum_{m=0}^{n} \eta \; r_s^n P_n^m(\cos\theta_s) \begin{Bmatrix} \cos m\varphi_s \\ \sin m\varphi_s \end{Bmatrix} \right\} \qquad 2.14$$

where $\eta = \left(2 - \delta_m^0\right)\frac{(n-m)!}{(n+m)!}$ $\quad \delta_m^0 \begin{cases} = 1 \; form = 0 \\ = 0 \; form \neq 0 \end{cases}$

and $P_n^m(\cos\theta)$ are associated Legendre coefficients

These infinite series can be avoided by using a closed form solution for the surface potential and this is introduced in the next section.

A more complex representation of the source is the quadrupole second moment tensor. This also is a point source representing a series of sources and sinks of infinite strength spatially separated by infinitely small distances but this representation is of pairs of dipoles with infinite dipole moment separated by infinitely small distances. The quadropole is (in the case of the electric potential) a symmetric dyadic tensor and has to be treated differently from a first order tensor (a vector):

$$\phi_\infty = \frac{1}{4\pi\sigma} \nabla . \tilde{Q} . \nabla \left(\frac{1}{R_e} \right) \qquad 2.15$$

The first set of quadrupoles will be derived. The 0th and 1st order terms of equation 2.15 are zero so taking the next term:

$$\tilde{Q} . \nabla \left(\frac{3(\bar{\bar{r}}\bar{r}_s)^2}{2r^5} - \frac{r_s^2}{2r^3} \right) \qquad 2.16$$

The term $\bar{\bar{r}}\bar{r}_s$ is the inner product of two vectors (1st) order tensors which give rise to the 2nd order tensor. The 'matrix' $\tilde{Q}$ effectively adds linear multipliers to this tensor but is referred to as the quadrupole tensor. The first part in brackets using dyadic notation:

$$(\bar{\bar{r}}\bar{\bar{r}}_s) = \begin{matrix} xx_s\hat{i}\hat{i} & xy_s\hat{i}\hat{j} & xz_s\hat{i}\hat{k} \\ yx_s\hat{j}\hat{i} & yy_s\hat{j}\hat{j} & yz_s\hat{j}\hat{k} \\ zx_s\hat{k}\hat{i} & zy_s\hat{k}\hat{j} & zz_s\hat{k}\hat{k} \end{matrix} \qquad 2.17$$

$$\tilde{Q}.\nabla(\bar{\bar{r}}\bar{\bar{r}}_s)^2 = \tilde{Q}.2\nabla(\bar{\bar{r}}\bar{\bar{r}}_s)\bar{r} = \begin{matrix} 2\left(Q_{xx}x^2x_s + Q_{xy}xyx_s + Q_{xz}xzx_s\right)\hat{i} \\ 2\left(Q_{yx}yxy_s + Q_{yy}y^2y_s + Q_{yz}yzy_s\right)\hat{j} \\ 2\left(Q_{zx}zxz_s + Q_{zy}zyz_s + Q_{zz}z^2z_s\right)\hat{k} \end{matrix} \qquad 2.18$$

$$\frac{3}{2r^5}\left[\nabla.\tilde{Q}.\nabla(\bar{\bar{r}}\bar{\bar{r}}_s)^2\right] = \frac{3}{r^5}\begin{vmatrix} (Q_{xx}x^2 + Q_{xy}xy + Q_{xz}xz + \\ Q_{yx}yx + Q_{yy}y^2 + Q_{yz}yz + \\ Q_{zx}zx + Q_{zy}zy + Q_{zz}z^2 \end{vmatrix} \qquad 2.19$$

and the second part of the brackets:

$$-\frac{1}{2r^5}\left[r^2\nabla.\tilde{Q}.\nabla\left(r_s^2\right)\right] = -\frac{1}{r^5}\left(Q_{xx}+Q_{yy}+Q_{zz}\right)\left(x^2+y^2+z^2\right) \qquad 2.20$$

These are assembled as spherical multipole components following the spherical harmonic expansion 2.14, so that 2.16:

$$\phi_2 = \frac{1}{4\pi\sigma r^5}\begin{vmatrix} -\frac{1}{2}A_{20}\left(x^2+y^2-2z^2\right) \\ +3A_{21}xz + 3B_{21}yz \\ +3A_{22}\left(x^2-y^2\right) + 6B_{22}xy \end{vmatrix}$$

but

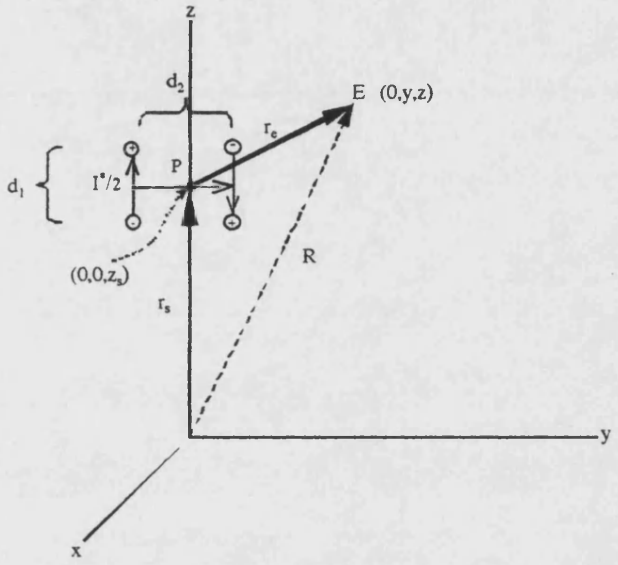
$$\begin{aligned} A_{20} &= -Q_{xx} - Q_{yy} + 2Q_{zz} \\ A_{21} &= Q_{xz} + Q_{zx} \qquad B_{21} = Q_{yz} + Q_{zy} \\ A_{22} &= \frac{1}{2}\left(Q_{xx} + Q_{yy}\right) \quad B_{22} = \frac{1}{2}\left(Q_{xy} + Q_{yx}\right) \end{aligned} \qquad 2.21$$

This can be compared with 2.13 and it is immediately obvious that the quadrupole is a set of dipole moments displaced by an infinitely small distance. This lowest order quadrupole contains no location information, only the higher terms do.

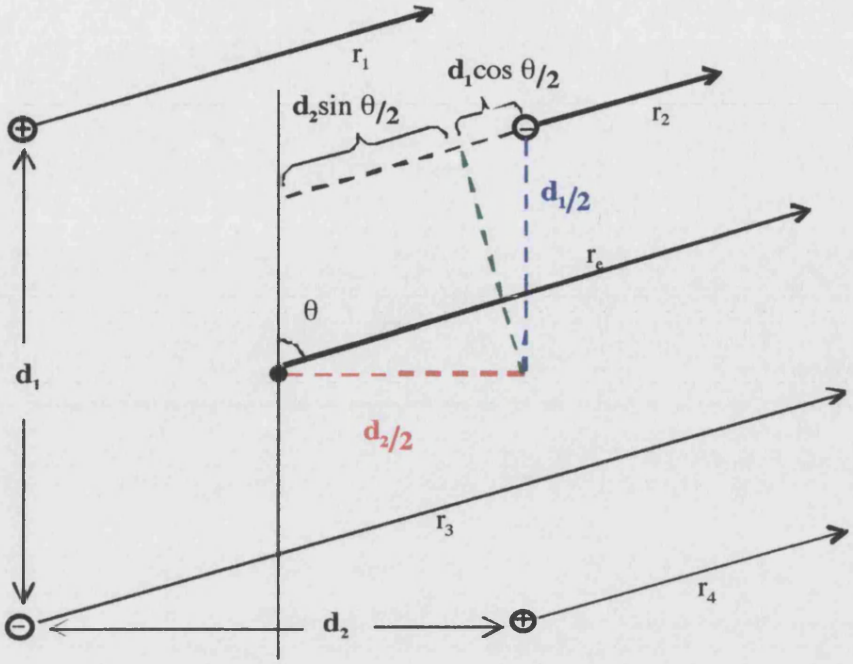
The result of the above analysis is that there are sets of infinite series, each series starting with a successively higher multipole located at the centre of the sphere and therefore containing no location information. In each case the location information is held in the higher order terms. It is quite possible to represent a surface potential map with a multpole series placed entirely at the origin and this is the usual analysis given in many papers (see for example [Wikswo and Swinney:1984]).

2.1 Geometric Interpretation of the Quadrupole

For illustrative purposes, a geometric interpretation of the quadrupole is presented. For brevity and simplicity one arrangement only is shown; opposing dipoles on the z axis displaced in the y direction:



The arrows between the point sources/sinks are the vector direction of displacement, d_1, of the + source and the vector direction of displacement , d_2, of the dipoles. E is an arbitrary point in the zy plane. The following diagrams show the zy distance differentials from each current source/sink:



From the above diagram and the symmetry of the opposing dipoles:

$$r_2 = r_e + \left(\frac{d_2}{2}\sin\theta + \frac{d_1}{2}\cos\theta\right)$$

$$r_3 = r_e - \left(\frac{d_2}{2}\sin\theta + \frac{d_1}{2}\cos\theta\right)$$

$$r_1 = r_e + \left(\frac{d_2}{2}\sin\theta - \frac{d_1}{2}\cos\theta\right)$$

$$r_4 = r_e - \left(\frac{d_2}{2}\sin\theta - \frac{d_1}{2}\cos\theta\right) \qquad 2.22$$

The potential at the field point E is from the super-position of the 4 sources:

$$(4\pi\sigma)\phi_p = \frac{I^*}{2}\left(\frac{1}{r_1} + \frac{1}{r_4} - \frac{1}{r_2} - \frac{1}{r_3}\right) \qquad 2.23$$

$$\left(\frac{1}{r_1} + \frac{1}{r_4} - \frac{1}{r_2} - \frac{1}{r_3}\right) = \frac{r_3 r_4(r_2 - r_1) + r_2 r_1(r_3 - r_4)}{r_1 r_2 r_3 r_4} \qquad 2.24$$

since $r_e^2 >> \left(\frac{d_2}{2}\sin\theta \pm \frac{d_1}{2}\cos\theta\right)^2$

$$r_1 r_2 r_3 r_4 \underset{d_1, d_2 \to 0}{\approx} r_e^4 \qquad 2.25$$

$$r_3 r_4 (r_2 - r_1) + r_2 r_1 (r_3 - r_4) = -2 r_e d_1 d_2 \sin\theta \cos\theta \qquad 2.26$$

with $r_e \sin\theta = y$ and $\cos\theta = \frac{1}{r_e}(z - z_s)$

$$(4\pi\sigma)\phi_p \approx -\frac{I^* d_1 d_2}{r_e^5} y(z - z_s) \qquad 2.27$$

$I^* d_1 d_2$ is the second moment of the source $\equiv Q_{zy}$

Equation 2.27 can be re-written as two terms, the quadrupole term that acts from the origin and contains no location information, and a term that is part of the next higher order. This is part of the B_{31} octapole term of the quadrupole expansion which does contain location information. Firstly, 2.27 is split:

$$(4\pi\sigma)\phi_p \approx -\left[\frac{Q_{zy}}{r_e^5} yz - \frac{Q_{zy} y z_s}{r_e^7}(x^2 + y^2 + z^2)\right] \qquad 2.28$$

(the – sign is due to the choice of polarity).

This is with respect to r_e which can be expanded to provide the remainder of the Q_{zy} component of B_{31} :

$$r_e = \sqrt{x^2 + y^2 + (z - z_s)^2}$$
$$R = \sqrt{x^2 + y^2 + z^2} \qquad 2.29$$

$$\frac{1}{r_e^5} = \frac{1}{R^5}\left(1 + \left\{\frac{-2zz_s + z_s^2}{R^2}\right\}\right)^{-\frac{5}{2}}$$
$$= \frac{1}{R^5}\left(1 - \frac{5}{2}\left\{\frac{-2zz_s + z_s^2}{R^2}\right\} + \ldots\ldots\right) \qquad 2.30$$

$$(4\pi\sigma)\phi_p \approx -\frac{Q_{zy}}{R^5} y \left[\begin{array}{l} z + \left(-\frac{5z}{2}\left\{\frac{-2zz_s}{R^2} + \frac{z_s^2(x^2 + y^2 + z^2)}{R^4}\right\} + \ldots\ldots\right) \\ - \left(\frac{Q_{zy} z_s}{R^2}(x^2 + y^2 + z^2) + \ldots\ldots\right) \end{array}\right] \qquad 2.31$$

$$(4\pi\sigma)\phi_p \approx -\frac{Q_{zy}}{R^5} yz - \left(\frac{Q_{zy} z_s y}{R^7}(x^2 + y^2 - 4z^2) + \ldots\ldots\right) \qquad 2.32$$

This is now with respect to the origin and contains the Q_{zy} quadrupole term plus the full Q_{zy} part of the B_{31} octapole term.

2.2 The Magnetic Field

The important theorem by Helmholtz states that the sources of a vector field contained inside some finite volume of space may be completely defined by the divergence and curl of that field. Proof of this theorem can be found in most textbooks on electromagnetics [eg Shadowitz:1975]. The irrotational and solenoidal parts of the field together give complete information about its source. In our case the magnetic field is the solenoidal part:

$$\bar{B}_\infty = \nabla \times \frac{\mu_0}{4\pi} \int_v \frac{\bar{j}_i}{R_e} dv \qquad 2.33$$

The second part of the right hand side is often referred to as the vector potential:

$$\bar{A}_\infty = \frac{\mu_0}{4\pi} \int_v \frac{\bar{j}_i}{R_e} dv \qquad 2.34$$

Although the vector potential has no physical significance, it is a useful mathematical concept when the equations have to be manipulated.

Using the vector identity:

$$\nabla \times (a\bar{\alpha}) = a\nabla \times \bar{\alpha} + (\nabla a) \times \bar{\alpha} \qquad 2.35$$

then 2.33 becomes:

$$\bar{B}_\infty = \frac{\mu_0}{4\pi} \int_v \left(\frac{1}{R_e}\right) \nabla \times \bar{j}_i \; dv + \frac{\mu_0}{4\pi} \int_v \nabla \left(\frac{1}{R_e}\right) \times \bar{j}_i \; dv \qquad 2.36$$

Using a modified Gauss' divergence theorem on the first term:

$$\int_v \nabla \times \bar{\alpha} \, dv = -\oint_s \bar{\alpha} \times d\bar{s} \qquad 2.37$$

since the closed surface integral of the source current is zero, only the second term is left:

$$\bar{B}_\infty = \frac{\mu_0}{4\pi} \int_v \nabla_p \left(\frac{1}{R_e}\right) \times \bar{j}_i \; dv \qquad 2.38$$

$$\text{or } -\frac{\mu_0}{4\pi}\int_v \nabla_p\left(\frac{1}{R_e}\right)\times \bar{j}_i \; dv \qquad 2.39$$

if the curl is taken at the field end of the R_e vector.

The treatment of this follows the same process as above resulting in the production of sets of infinite series the first one being the dipole series:

$$\begin{aligned}
B_{x1} &= -\frac{\mu_0}{4\pi r^3}(D_z y - D_y z)\\
B_{y1} &= -\frac{\mu_0}{4\pi r^3}(D_x z - D_z x)\\
B_{z1} &= -\frac{\mu_0}{4\pi r^3}(D_y x - D_x y)
\end{aligned} \qquad 2.40$$

$$\begin{aligned}
B_{x2} &= \frac{\mu_0}{4\pi r^5}\begin{Bmatrix}-3(xyD_x z_s + zyD_z z_s)+3(zxD_x y_s + zyD_y y_s)\\ -\left(2y^2 - x^2 - z^2\right)D_y z_s + \left(2z^2 - x^2 - y^2\right)D_z y_s\end{Bmatrix}\\
B_{y2} &= \frac{\mu_0}{4\pi r^5}\begin{Bmatrix}-3(yzD_y x_s + xzD_x x_s)+3(xyD_y z_s + xzD_z z_s)\\ -\left(2z^2 - y^2 - x^2\right)D_z x_s + \left(2x^2 - z^2 - y^2\right)D_x z_s\end{Bmatrix}\\
B_{z2} &= \frac{\mu_0}{4\pi r^5}\begin{Bmatrix}-3(zxD_z y_s + yxD_y y_s)+3(yzD_z x_s + yxD_x x_s)\\ -\left(2x^2 - y^2 - z^2\right)D_x y_s + \left(2y^2 - x^2 - z^2\right)D_y x_s\end{Bmatrix}
\end{aligned} \qquad 2.41$$

and the second one being the quadrupole series, the first term of which:

$$\begin{aligned}
B_{x2} &= \frac{\mu_0}{4\pi r^5}\begin{Bmatrix}-3(xyQ_{xz} + zyQ_{zz})+3(zxQ_{xy} + zyQ_{yy})\\ -\left(2y^2 - x^2 - z^2\right)Q_{yz} + \left(2z^2 - x^2 - y^2\right)Q_{zy}\end{Bmatrix}\\
B_{y2} &= \frac{\mu_0}{4\pi r^5}\begin{Bmatrix}-3(yzQ_{yx} + xzQ_{xx})+3(xyQ_{yz} + xzQ_{zz})\\ -\left(2z^2 - y^2 - x^2\right)Q_{zx} + \left(2x^2 - z^2 - y^2\right)Q_{xz}\end{Bmatrix}\\
B_{z2} &= \frac{\mu_0}{4\pi r^5}\begin{Bmatrix}-3(zxQ_{zy} + yxQ_{yy})+3(yzQ_{zx} + yxQ_{xx})\\ -\left(2x^2 - y^2 - z^2\right)Q_{xy} + \left(2y^2 - x^2 - z^2\right)Q_{yx}\end{Bmatrix}
\end{aligned} \qquad 2.42$$

This contains the electric multipole components $\begin{Bmatrix}A_{nm}\\ B_{nm}\end{Bmatrix}$ but also three non-vortex sources which are the anti-symmetric parts of the quadrupole tensor, since every second rank tensor can be resolved into symmetric and anti-symmetric parts through using the identity:

$$Q_{\alpha\beta} = \frac{1}{2}\left(Q_{\alpha\beta} + Q_{\beta\alpha}\right) + \frac{1}{2}\left(Q_{\alpha\beta} - Q_{\beta\alpha}\right) \qquad 2.43$$

In the case of the electric multipole components $Q_{\alpha\beta} = Q_{\beta\alpha}$ so that any anti-symmetric components cancel out and are electrically neutral. Equation 2.42 can be re-written to show the contribution to the magnetic field from the symmetric and anti-symmetric quadrupole components. Only B_{x2} is shown as an example:

$$B_{x2} = \frac{1}{4\pi\sigma r^5}\left[-\frac{3}{2}A_{20}\left(x^2 + y^2 - 2z^2\right) + 3A_{21}xz + 3B_{21}yz + 3A_{22}yz + B_{22}xz\right] + \frac{1}{4\pi\sigma r^5}\left[3A_{10}^M xz + A_{11}^M\left(2x^2 - y^2 - z^2\right) - 3B_{11}^M xy\right] \qquad 2.44$$

where $\begin{Bmatrix} A_{nm}^M \\ B_{nm}^M \end{Bmatrix}$ are the lowest order magnetic multipole components, the magnetic dipoles:

$$A_{10}^M = \frac{1}{2}\left(Q_{xy} - Q_{yx}\right)$$
$$A_{11}^M = \frac{1}{2}\left(Q_{yz} - Q_{zy}\right) \qquad B_{11}^M = \frac{1}{2}\left(Q_{zx} - Q_{xz}\right) \qquad 2.45$$

Again, it is noted that this lowest order expression in the quadrupole series contains no location information, only the higher order terms (not shown).

2.3 Conclusion

The approach to the development of multipole series developed in this section shows that it is possible to model separate arbitrarily located point sources. By including higher terms in the multipole series and starting a series at a higher term, it is possible to define one of the sources as being either a solitary dipole or a solitary quadrupole. There is no physical interpretation of either of these although in the case of the dipole there appears to be closer analogy to the cellular ionic mechanism. The dipole is easier to picture and easier to handle mathematically which partly explains its common usage although, as described in the introduction, it would appear to be inappropriate for re-entry and circulatory action potential behaviour. The mathematics of representing higher order sources is certainly more involved and the next section develops a closed form for representing arbitrarily located point sources in a homogeneous sphere. With such a closed form and using the approach above, it is possible to represent the source as combinations of dipoles and quadrupoles which could be either separate or coincident. The location information in the equations is non-linear but the source magnitudes (the

multipoles) are linear. An efficient linear/non-linear least squares algorithm will be required for an inverse model and this is described in section 5.

As a final note, the total magnetic field is influenced by the conductivity change at the sphere boundary and this is incorporated by modelling the boundary as a hypothetical dipole current layer in the infinite homogeneous medium and placed at the boundary radius. The mathematics were developed by [Geselowitz:1970] and are used in the next section. This adjustment to the infinite medium magnetic field is due to the behaviour of the ohmic (or volume) currents at the boundary and involves the surface integration of this hypothetical dipole source. As the dipole layer has been created in the first instance by a point source, then the integration expression contains location information. There is no closed form solution for the integration, so a dipole infinite series expression results and sufficient terms are then included to give the required accuracy for this adjustment to the magnetic field.

3. Development of Generalised Multipole Equations for the Bounded Sphere

The accurate location of one or more point current sources in a homogeneous volume conductor from surface data measurements requires the development of equations which can account for changes in the strength and direction of the vector components of the sources together with changes in the position of those sources. These equations are called the forward model and must predict the electric potential and the magnetic field measurements at any one surface point from a knowledge of this data. The inverse model is simply the forward model repeatedly run with successive small changes in input data to minimise the difference between the computed and actual measurements at the surface. In developing these equations there are a number of requirements.

The electric potential is a scalar measurement and is acquired by the attachment to the patient's skin of electrode arrays. Data interpolation is used for areas of sparse coverage. The electrode positions are identified in relation to known three dimensional landmarks in the body. The magnetic field is a vector measurement and the sensing coils must be orientated in a known plane. These planes are usually the orthogonal cartesian planes but could be radial. The requirement, therefore, is to develop equations where all locations and all vector directions are in cartesian (x,y,z) co-ordinates. This will also allow the testing of the inverse model accuracy on a plane-by-plane basis and the improvement, if any, of two or more planes. The equations in the following development often are in spherical (r,θ,ϕ) co-ordinates or mathematical operators (eg ∇ the nabla operator) are best applied in spherical form. This necessitates a certain amount of conversion between co-ordinate systems and the following standard unit vector relationships are applied :

$$\hat{U}_R = \hat{U}_x \sin\theta\cos\varphi + \hat{U}_y \sin\theta\sin\varphi + \hat{U}_z \cos\theta \qquad 3.1a$$

$$\hat{U}_\theta = \hat{U}_x \cos\theta\cos\varphi + \hat{U}_y \cos\theta\sin\varphi - \hat{U}_z \sin\theta \qquad 3.1b$$

$$\hat{U}_\varphi = -\hat{U}_x \sin\varphi + \hat{U}_y \cos\varphi \qquad 3.1c$$

$$\hat{U}_x = \hat{U}_R \sin\theta\cos\varphi + \hat{U}_\theta \cos\theta\cos\varphi - \hat{U}_\varphi \sin\varphi \qquad 3.2a$$

$$\hat{U}_y = \hat{U}_R \sin\theta\sin\varphi + \hat{U}_\theta \cos\theta\sin\varphi + \hat{U}_\varphi \cos\varphi \qquad 3.2b$$

$$\hat{U}_z = \hat{U}_R \cos\theta - \hat{U}_\theta \sin\theta \qquad 3.2c$$

where $\hat{U}_\alpha$ is a unit vector in co-ordinate plane α.

Extending the equations from the dipole case to the quadrupole case involves a differentiation with respect to each co-ordinate plane. Dipole equations are likely therefore to treble in complexity and in number of terms. It would be preferable if closed form rather than infinite series equations are used in order to minimise the inverse computing time. Section 3.1 describes the closed form expression for the spatial gradient (the lead field) at any point inside a sphere in relation to any point at its surface. This equation is due to [Brody et al 1973] and is extended from the dipole case to cover higher order multipole sources and also to the infinite medium magnetic field. The magnetic field also needs to be adjusted for the influence of the finite conductivity boundary at the surface and this involves a surface integration for every surface point evaluated. The integration on the closed form results in an infinite series so that no advantage is gained by this approach. Instead, section 3.3 solves this integration using an infinite series from the outset.

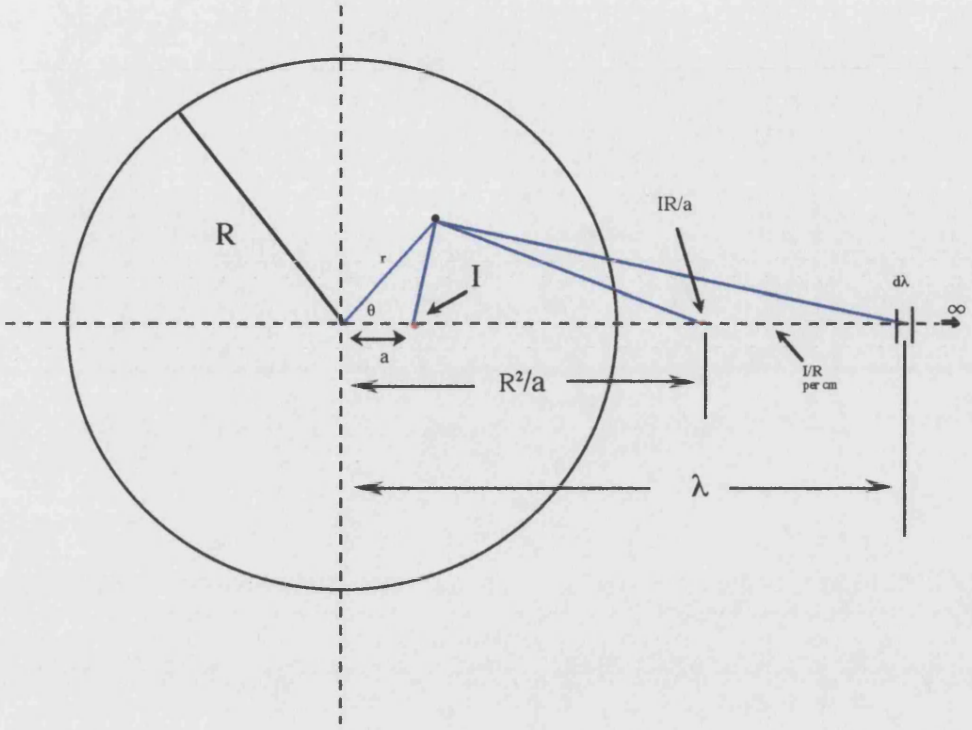
The following sections develop the equations for the quadrupole case and, due to the switching between co-ordinate systems, a well defined notational system is used. This proved to be very important in the computer modelling stage as very small errors in the formulae led to very large errors in the resultant signal. The least squares inverse solution requires a further stage in differentiation in order to set the direction for the subsequent iteration and, effectively, the 27 octopole infinite series terms have to be computed. This also emphasises the importance of a rigorous notation system.

3.1 The Arbitrarily Located Dipole

The closed form expression described here was developed by Brody [*op. cit.*] who defined the lead field (or spatial gradient) at a point P inside a sphere, generated by a point of interest on its surface, as a scalar potential function, U, whose (first order) gradient at P is the lead vector $\overline{F}$. The potential, V, on the surface due to a dipole source, $\overline{D}$ is then:

$$V = \overline{D} \cdot \overline{F} \qquad 3.3$$

It is worth summarising the derivation of the scalar potential function as this forms the part of the basis of this thesis. By definition, the scalar function, U, is generated when the surface point is reciprocally energized with one unit of current [McFee et al: 1952]. A unit current source is placed arbitrarily in a homogeneous conducting sphere, of radius R, and a line is drawn from the centre of the sphere through this source and on to infinity. The image line of this source is the radius line passing through I and on to infinity. The image point is located on this line and the line has a current density of I/R per cm. The distance, d , of the image point from the surface satisfies: $\frac{I}{(R-a)} = \frac{(R+d)I}{d.R}$ which gives the distance from the origin to the image point of $\frac{R^2}{a}$ and an image strength of IR/a:



The standard textbook solution to the resulting Legendre Equations for the potentials generated by the source is expressed in Legendre polynomials. It is made up of three parts:

a) due to the source (r>a)

$$V_s = \frac{I}{4\pi\sigma}\sum_{n=0}^{\infty}\frac{a^n}{r^{n+1}}P_n\cos\theta \qquad 3.4$$

b) due to the image ($r<R^2/a$)

$$V_i = \frac{1}{4\pi\sigma}\sum_{n=0}^{\infty}\left(\frac{IR}{a}\right)\frac{r^n}{\left(R^2/a\right)^{n+1}}P_n(\cos\theta) \qquad 3.5$$

c) due to any element of the image line

$$dV_{il} = \frac{I\,d\lambda}{4\pi\sigma R}\sum_{n=0}^{\infty}\frac{r^n}{\lambda^{n+1}}P_n(\cos\theta) \qquad 3.6$$

where σ is the specific conductivity of the medium.

These expressions may now be differentiated with respect to r, and summed at r=R to give the total normal component of the gradient at the spherical boundary resulting in:

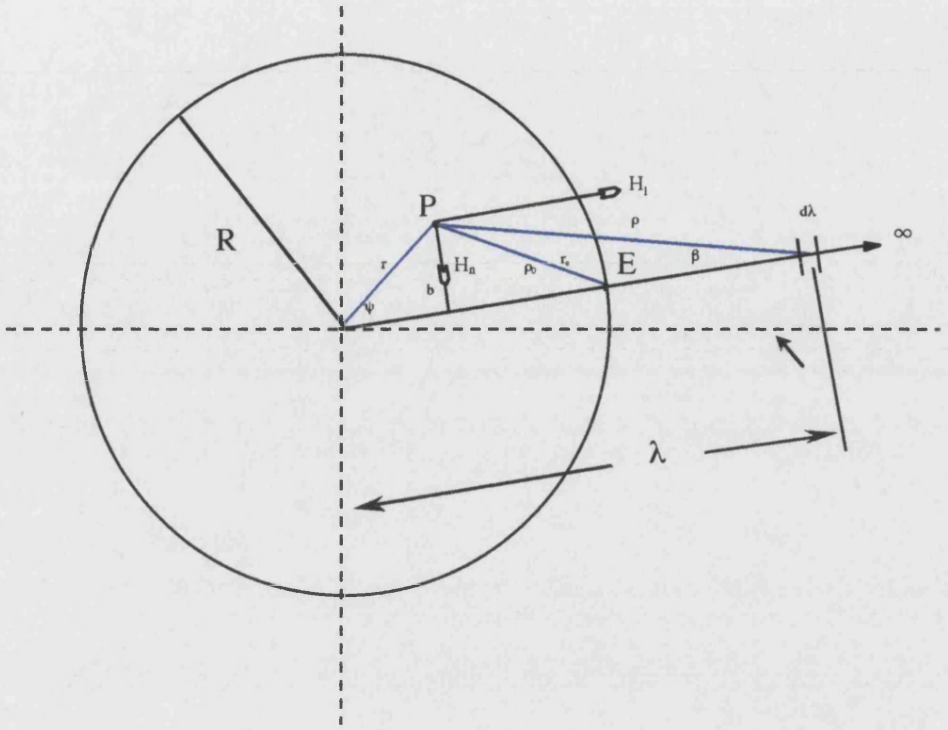
$$\left[\frac{\partial V_s}{\partial r}\right]_{r=R} = -\frac{I}{4\pi\sigma}\left[\frac{1}{R^2} + \sum_{n=1}^{\infty}(n+1)\frac{a^n}{R^{n+2}}P_n\cos\theta\right] \qquad 3.7$$

$$\left[\frac{\partial V_i}{\partial r}\right]_{r=R} = \frac{I}{4\pi\sigma}\left[\sum_{n=1}^{\infty} n\frac{a^n}{R^{n+2}}P_n\cos\theta\right] \qquad 3.8$$

$$\int_{R^2/a}^{\infty}\left[\frac{\partial}{\partial r}dV_{ll}\right]_{r=R} d\lambda = \frac{I\,d\lambda}{4\pi\sigma}\sum_{n=1}^{\infty}\frac{a^n}{R^{n+2}}P_n(\cos\theta) \qquad 3.9$$

The total normal component of the gradient is therefore $\left(-\frac{I}{4\pi\sigma R^2}\right)$

If the source, I, is set to unity (the lead field definition) and moved to the boundary, the image and the source then coincide forming a two unit strength source with an image line extending from the surface to infinity:



The field gradient at P owing to the double source is

$$\bar{F} = \frac{2}{4\pi\sigma\rho_0^2}\hat{U}_{\rho_0} \qquad 3.10$$

plus two components (normal, H_n , and parallel, H_l) owing to the image line:

$$H_n = \frac{1}{4\pi\sigma R}\int_R^\infty \frac{\sin\beta}{\rho^2}d\lambda \qquad 3.11$$

$$H_l = \frac{1}{4\pi\sigma R}\int_R^\infty \frac{\cos\beta}{\rho^2}d\lambda \qquad 3.12$$

Evaluating these, for example, in the x direction and, after making a substitution to allow for the case that b=0, the source at the centre of the sphere gives:

$$F_x = \frac{1}{4\pi\sigma\rho_0}\left(\left[\frac{2(x_e - x)}{\rho_0^2}\right] + \frac{1}{R^2}\left[x_e + \frac{x_e r\cos\psi - Rx}{\rho_0 + R - r\cos\psi}\right]\right) \qquad 3.13$$

where point E is (x_e, y_e, z_e) and point P is (x,y,z)

$$\rho_0^2 = (x_e - x)^2 + (y_e - y)^2 + (z_e - z)^2 \qquad r\cos\psi = \frac{x_e x + y_e y + z_e z}{R} = \gamma$$

The full expression for the potential on the outside of a sphere from an arbitrarily located dipole is:

$$V = D_x F_x + D_y F_y + D_z F_z \qquad 3.14$$

Moving the dipole to the centre of the sphere (x,y,z) = 0 reduces the expression to:

$$V = \frac{3\left(D_x x_e + D_y y_e + D_z z_e\right)}{4\pi\sigma R^3}$$

This expression, without the factor 3, is familiar from textbooks for the field generated by a dipole at the centre of an unbounded sphere and, for example, can be seen in spherical form in [Geselowitz:1960] . The effect of the boundary has been evaluated by [Geselowitz and Ishiwatari:1965] who applied the boundary conditions at the sphere surface and solved the resulting Laplacian equation in spherical co-ordinates for the general case of an eccentric multipole. They showed that the correction to the unbounded spherical harmonic expression for the potential to be a factor $\frac{2n+1}{n}$ where n is the multipole order. For the dipole case n has the value 1 and hence the resulting factor 3.

3.2 The Eccentric Quadrupole

As shown in the previous chapter the potential arising from an arbitrarily located quadrupole is

$$\frac{1}{4\pi\sigma}\nabla_p \cdot \tilde{Q} \cdot \nabla_p\left(\frac{1}{r_e}\right) \qquad 3.15$$

where r_e is the line joining the source P with the surface point of measurement E.

This is the second term of the Taylor series expansion of the general multipole expression:

$$\frac{1}{2}\frac{1}{4\pi\sigma}\nabla_p \cdot \tilde{Q} \cdot \nabla_p\left(\frac{1}{r_e}\right)$$

which may be seen in [Geselowitz:1965]. The $\frac{1}{2}$ arises from quadrupole symmetry and will be ignored as it is will merely be subsumed into the resulting quadrupole vector strengths. The sign is arbitrary and is also subsumed into the quadrupole vector directions. The del operator acts at the source end of the space vector which points from the source to the surface. It should be noted that the del operator can be applied to the other end of the space vector with the same result but reversed sign.

$\tilde{Q}$ the quadrupole tensor is the second moment of the impressed source current about its origin. $\tilde{Q}$ is a point source and its individual elements are defined as:

$$Q_{ij} = \int_v J_i x_i x_j \, dv \Big|_{x_i x_j \to 0,\ J_i \to \infty} \qquad 3.16$$

where v refers to the smallest spherical volume enclosing the source J_i

and where $x_{i,j} = x, y, z$

The dipole expression above for the surface potential is $\bar{D} \cdot \bar{F} = \frac{3}{4\pi\sigma}\bar{D} \cdot \nabla_p\left(\frac{1}{r_e}\right)$ so by simple extension the surface potential due to a point quadrupole is:

$$V = \nabla_p \cdot \tilde{Q} \cdot \bar{F} \qquad 3.17$$

noting that this will contain a factor $\left.\frac{2n+1}{n}\right|_{n=2}$

Expanding this gives:

$$\left(\begin{array}{l} \frac{\partial}{\partial x}\left\{F_x Q_{xx} + F_y Q_{yx} + F_z Q_{zx}\right\} \\ + \frac{\partial}{\partial y}\left\{F_x Q_{xy} + F_y Q_{yy} + F_z Q_{zy}\right\} \\ + \frac{\partial}{\partial z}\left\{F_x Q_{xz} + F_y Q_{yz} + F_z Q_{zz}\right\} \end{array}\right) \qquad 3.18$$

As has been pointed by [Geselowitz:1965] there are only 5 independent components of this tensor since:

$$Q_{xy} = Q_{yx},\ Q_{xz} = Q_{zx},\ Q_{xz} = Q_{zx} \text{ and } Q_{xx} + Q_{yy} + Q_{zz} = 0 \qquad 3.19$$

There are two differential types to evaluate:

$$\frac{\partial}{\partial x_i} F_{x_i} \quad \text{and} \quad \frac{\partial}{\partial x_j} F_{x_i} \qquad 3.20$$

where:

$$F_{x_i} = \frac{1}{4\pi\sigma\rho_0}\left(\left[\frac{2\left(x_{e_i} - x_i\right)}{\rho_0^2}\right] + \frac{1}{R^2}\left[x_{e_i} + \frac{x_{e_i}\gamma - Rx_i}{\rho_0 + R - \gamma}\right]\right) \qquad 3.21$$

resulting in:

$$\frac{\partial}{\partial x_j}\left[\frac{2\left(x_{e_i} - x_i\right)}{\rho_0^3}\right] = -\frac{2}{\rho_0^3}\delta_{x_j}^{x_i} + \frac{6\left(x_{e_i} - x_i\right)^2}{\rho_0^5} \qquad \frac{\partial}{\partial x_j}\left[\frac{x_{e_i}}{R^2\rho_0}\right] = \frac{x_{e_i}\left(x_{e_j} - x_j\right)}{R^2\rho_0^3} \qquad 3.22$$

$$\frac{\partial}{\partial x_j}\left[\frac{x_{e_i}\gamma - Rx_i}{R^2\rho_0\left(\rho_0+R-\gamma\right)}\right] = \frac{\frac{x_{e_i}x_{e_j}}{R} - R\delta_{x_j}^{x_i}}{R^2\rho_0\left(\rho_0+R-\gamma\right)} + \frac{\left(x_{e_i}\gamma - Rx_i\right)}{R^2}\frac{\left(x_{e_j}-x_{e_j}\right)\left(2\rho_0+R-\gamma\right)+\frac{\rho_0^2 x_{e_j}}{R}}{\rho_0^3\left(\rho_0+R-\gamma\right)}$$

3.23

in which $\delta_{x_j}^{x_i}$ is the Kroneker delta function.

As a check on the above, after applying this differential to all the combinations and setting (x,y,z)=(0,0,0) ie the quadrupole at the centre of the sphere gives:

$$V = \frac{5}{2}\frac{1}{4\pi\sigma R}\left[\begin{array}{l}\frac{1}{2}\left(Q_{xx}+Q_{yy}-2Q_{zz}\right)\left(x_e^2+y_e^2-2z_e^2\right) + \frac{3}{2}\left(Q_{xx}-Q_{yy}\right)\left(x_e^2-y_e^2\right) \\ + 3\left(Q_{xy}+Q_{yx}\right)\left(x_e y_e\right) + 3\left(Q_{xz}+Q_{zx}\right)\left(x_e z_e\right) + 3\left(Q_{yz}+Q_{zy}\right)\left(y_e z_e\right)\end{array}\right]$$

3.24

Leaving out the boundary factor of $\frac{5}{2}$ gives the infinite medium potential which can be found in many papers, for example [Katila and Karp:1983].

The same expression may be derived in spherical harmonics for a source confined to the co-ordinate origin as shown, for example, by [Titomir and Kneppo:1994]

$$V(r,\theta,\varphi) = \frac{1}{4\pi\sigma}\sum_{n=0}^{\infty}\sum_{m=0}^{n}\frac{1}{R^{n+1}}\left[A_{nm}P_n^m(\cos\theta)\cos m\varphi + B_{nm}P_n^m(\cos\theta)\sin m\varphi\right] \qquad 3.25$$

where A_{nm} and B_{nm} are the coefficients of the even and odd harmonics at radius R.

Expanding this for the quadrupole case of n=2 yields:

$$V = \frac{5}{2}\frac{1}{4\pi\sigma R}\left[\begin{array}{l}-\frac{1}{2}A_{20}\left(x_e^2+y_e^2-2z_e^2\right) + 3A_{22}\left(x_e^2-y_e^2\right) + 6B_{22}\left(x_e y_e\right) \\ + 3A_{21}\left(x_e z_e\right) + 3B_{21}\left(y_e z_e\right)\end{array}\right] \qquad 3.26$$

Thus the quadrupole coefficients are:

$$A_{20} = \left(-Q_{xx} - Q_{yy} + 2Q_{zz}\right) \quad A_{21} = \left(Q_{xz} + Q_{zx}\right) \quad A_{22} = \frac{1}{2}\left(Q_{xx} - Q_{yy}\right)$$
$$B_{21} = \left(Q_{yz} + Q_{zy}\right) \quad B_{22} = \frac{1}{2}\left(Q_{xy} + Q_{yx}\right) \qquad 3.27$$

3.3 The Eccentric Octapole

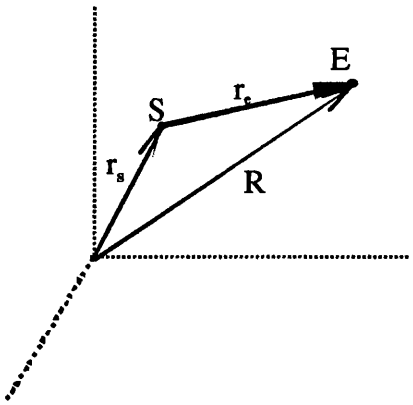
The octapole tensor is the third order moment of the impressed current source about its origin and gives rise to a surface potential as follows:

$$V = (\nabla_p \,.\, \tilde{O} \,.\, \nabla_p).\left(F_x F_y F_z\right) \qquad 3.28$$

It consists of 27 terms $\frac{\partial}{\partial x_{i,j}}\left[\frac{\partial}{\partial x_{i,j}}\right] F_{x_{i,j}}$ of which 7 are independent. The result of this differentiation can be seen in the model coding in Appendix 8 where it is required in the least squares inverse model for the quadrupole.

3.4 The Infinite Medium Magnetic Field

As described in Chapter 2, the magnetic field may be derived from the magnetic vector potential, $\bar{A}(\bar{r})$, generated by the current source. In an infinite homogeneous medium this potential is directed from the source to the field measurement point :



The magnetic field is then given by the curl of this vector potential at the field point:

$$\bar{B}(\bar{r}_e) = \nabla_s \times_s \bar{A}(\bar{r}_e) = \nabla_e \times_e \bar{A}(\bar{r}_e) \qquad 3.29$$

3.4.1 The Dipole Source

Due to a point dipole source the magnetic vector potential in a homogeneous medium is simply:

$$\bar{A}(\bar{r}_e) = \frac{\mu_0}{4\pi}\left(\frac{\bar{D}}{r_e}\right) \qquad 3.30$$

and the curl of this reduces to:

$$\bar{B}(\bar{r}_e) = \frac{\mu_0}{4\pi}\nabla_s\left(\frac{1}{r_e}\right)\times_s \bar{D} = -\frac{\mu_0}{4\pi}\nabla_s\left(\frac{1}{r_e}\right)\times_e \bar{D} \qquad 3.31$$

The missing term, $\frac{1}{r_e}(\nabla_e \times \bar{D})$ is transformed by Green's Theorem and integrates to zero (see, for example, [Titomir and Kneppo:1994]).

Since this expression is based on the vector potential, the lead field derivation above may be applied with minor changes. Firstly, the medium conductivity is replaced by its permeability and secondly, the factor for the boundary correction needs to be excluded since we are dealing here with an infinite medium potential. Thus:

$$\bar{B}(\bar{r}_e) = -\frac{\mu_0}{4\pi}\frac{1}{3}(\bar{F}\times\bar{D}) \qquad 3.32$$

and, for example:

$$\bar{B}_{D_x} = -\frac{\mu_0}{4\pi}\frac{1}{3\rho_0}\begin{bmatrix} D_z\left(\frac{2(y_e-y)}{\rho_0^2}\right)+\frac{D_z}{R^2}\left\{y_e+\frac{y_e\gamma - Ry}{\rho_0+R-\gamma}\right\} \\ -D_y\left(\frac{2(z_e-z)}{\rho_0^2}\right)+\frac{D_y}{R^2}\left\{z_e+\frac{z_e\gamma - Rz}{\rho_0+R-\gamma}\right\} \end{bmatrix} \qquad 3.33$$

with the source at the centre of the sphere this reduces to:

$$\bar{B}_{D_x} = \frac{\mu_0}{4\pi R^3}\left(D_y z_e - D_z y_e\right) \qquad 3.34$$

3.4.2 The Quadrupole Source

A direct analogy with the scalar potential yields the magnetic vector potential produced by the point quadrupole source. As described by [Katila and Karp:1983] the dipole source for the scalar potential is the zero[th] order moment for the vector potential. The first order moment contribution to the vector potential would therefore be:

$$\begin{aligned}\bar{A}(\bar{r}_e) &= \frac{\mu_0}{4\pi}\nabla_e \cdot\left|\frac{\tilde{Q}}{r_e}\right| = -\frac{\mu_0}{4\pi}\nabla_s \cdot\left|\frac{\tilde{Q}}{r_e}\right| \\ &= -\frac{\mu_0}{4\pi}\left\{\tilde{Q}\cdot\nabla_s\left(\frac{1}{r_e}\right)\right\} = -\frac{\mu_0}{4\pi}\frac{2}{5}\left(\tilde{Q}\cdot\bar{F}\right)\end{aligned} \qquad 3.35$$

Using tensor arithmetic, the source term and lead field product yields the 3 element vector:

$$\left\{\left(F_x Q_{xx} + F_y Q_{yx} + F_z Q_{zx}\right)\hat{i},\ \left(F_x Q_{xy} + F_y Q_{yy} + F_z Q_{zy}\right)\hat{j},\ \left(F_x Q_{xz} + F_y Q_{yz} + F_z Q_{zz}\right)\hat{k}\right\} \qquad 3.36$$

The magnetic field is then obtained from the curl of this:

$$\bar{B}_Q = -\frac{\mu}{4\pi}\frac{2}{5}\left[\nabla_e \times\left(\tilde{Q}\cdot\bar{F}\right)\right] = \frac{\mu}{4\pi}\frac{2}{5}\left[\nabla_s \times\left(\tilde{Q}\cdot\bar{F}\right)\right] \qquad 3.37$$

Using the notation $F_{\alpha\beta} = \frac{\partial}{\partial\beta}F_\alpha$ then the x component of the field, for example , is

$$B_{Q_x} = \frac{\mu}{4\pi}\frac{2}{5}\left\{\left(F_{xy}Q_{xz} + F_y y Q_{yz} + F_{zy}Q_{zz}\right) - \left(F_{xz}Q_{xy} + F_{yz}Q_{yy} + F_{zz}Q_{zy}\right)\right\} \qquad 3.38$$

Again, details of this can be seen in the coding in Appendix 8. However, the same exercise of moving the source back to the origin (x,y,z)=(0,0,0) yields:

$$B_{Q_x} = \frac{\mu}{4\pi R^5}\begin{Bmatrix} -3\left(x_e y_e Q_{xz} + z_e y_e Q_{zz}\right) + 3\left(z_e x_e Q_{xy} + z_e y_e Q_{yy}\right) \\ -\left(2y_e^2 - x_e^2 - z_e^2\right)Q_{yz} + \left(2z_e^2 - x_e^2 - y_e^2\right)Q_{zy} \end{Bmatrix} \qquad 3.39$$

A similar expression, but for the B_z, term can be found in [Nenonen,Katila et al:1991].

A tensor can be written as the sum of its symmetric and antisymmetric parts and in the case of the quadrupole:

Symmetric	*Antisymmetric*
$Q_{ij} = Q_{ji}\Big\vert_{i \neq j}$	$Q_{ij} + Q_{ji} \neq 0\Big\vert_{i \neq j}$

Splitting B_{Q_x} into its component parts and using the electric quadrupole coefficients defined above gives the symmetric part as:

$$B_{Q_x} = \frac{\mu}{4\pi R^5}\left\{-\frac{3}{2}A_{20}y_e z_e - \frac{3}{2}A_{21}x_e z_e - \frac{3}{2}B_{21}\left(y_e^2 - z_e^2\right) - 3A_{22}y_e z_e + B_{22}x_e z_e\right\} \qquad 3.40$$

Thus the symmetric quadrupole coefficients give rise to both an electric potential and a magnetic field . However, the antisymmetric coefficients give rise only to a magnetic field and are expressed in terms of magnetic dipole coefficients, defined as:

$$A_{10}^M = \frac{1}{2}\left(Q_{xy} - Q_{yx}\right) \quad A_{11}^M = \frac{1}{2}\left(Q_{yz} - Q_{zy}\right) \quad B_{11}^M = \frac{1}{2}\left(Q_{zx} - Q_{xz}\right)$$

$$B_{Q_x}(anti) = \frac{\mu}{4\pi R^5}\left\{3A_{10}^M x_e y_e + 3A_{11}^M\left(2x_e^2 - y_e^2 - z_e^2\right) + 3B_{11}^M x_e y_e\right\} \qquad 3.41$$

3.5 The Influence of the Surface Boundary

The volume currents flow radially from the source throughout the homogeneous volume and are proportional to the gradient of the scalar potential created by the source. Since the curl of the gradient of a scalar is zero, the Coulombian currents do not give rise to a magnetic field. However, in a bounded conductor the volume currents are deflected at the conductivity interface and, as shown by [Geselowitz:1970], give rise to a magnetic field at this point. These volume current sources are treated as a double layer current source, a distributed dipole source, on the surface of the sphere and the magnetic field at any point on the surface is the integral sum of the field generated by the double layer across the whole surface as seen by that point. In terms of the magnetic vector potential:

$$\bar{A}(\bar{r}) = \frac{\mu}{4\pi}\int_v \frac{\bar{J}^*}{r_e} dv - \frac{\mu}{4\pi}\oint_S \frac{\sigma V}{R_p}\,\bar{dS} \qquad 3.42$$

where R_p is the line from any surface point to the point of evaluation, also on the surface. The first term is the infinite medium magnetic potential as described above and the second term is the correction to this potential due to the boundary.
Using the closed form expressions developed above for the electric potential, V, on the surface of a sphere does not lead to a finite series after the surface integration. Therefore, the approach taken here is to use the spherical harmonic expansion for the electric potentials developed by a point multipole source and then carry out the surface integration. This is a lengthy procedure and details are given in Appendix 1. This method is a generalisation of a special case pointed out by [Grynszpan:1971].

3.5.1 Notation

In what follows, care has to be taken with notation as the method uses mixed spherical and rectangular systems with appropriate swapping to facilitate the mathematics:

Subscripts

n,m,n',m'	summation indices
e	surface field measurement point
s	source point
unscripted	any surface point
x,y,z	)when used with F:
r,θ,ψ	)vector differentiation {ie scale factors) included}with respect to one or more) co-ordinates in the order given by the)subscripts.)otherwise: a vector component

$P_n^m(\,)$ Associated Legendre polynomial

λ_{nm} the elements of the harmonic expansion for V which relate to the source

F the elements of λ which are differentiable with respect to the source co-ordinates

δ_α^β Kroneker delta function: $\delta_\alpha^\beta \begin{cases} =1 & for\ \alpha=\beta \\ =0 & for\ \alpha\neq\beta \end{cases}$

η index dependent constant $\eta = \left(2-\delta_m^0\right)\dfrac{(n-m)!}{(n+m)!}$

$\hat{u}_\alpha$ a unit vector in the direction α

3.5.2 The Spherical Harmonic Series

The electric potential at any point on the surface due to a point multipole is inversely proportional to the gradient of the space vector pointing from the source to the field point on the surface. The expansion of this gradient is termed a multipole series. Thus, for the dipole moment:

$$V = \frac{1}{4\pi\sigma}\bar{D} \cdot \nabla\left(\frac{1}{r_p}\right) \qquad 3.43$$

and for the quadrupole tensor moment:

$$V = \frac{1}{4\pi\sigma}\nabla \cdot \tilde{Q} \cdot \nabla\left(\frac{1}{r_p}\right) \qquad 3.44$$

This expansion allows separation of the source and field related elements:

$$V = \frac{1}{4\pi\sigma}\sum_{n=1}^{\infty}\sum_{m=0}^{n}\frac{2n+1}{n}\frac{\lambda_{nm}}{R^{n+1}}P_n^m(\cos\theta)\begin{Bmatrix}\cos m\varphi\\ \sin m\varphi\end{Bmatrix} \qquad 3.45$$

where, for example, λ for the dipole term is:

$$\lambda_{nm} = \bar{D} \cdot \nabla_{r_s}\left\{\sum_{n=1}^{\infty}\sum_{m=0}^{n}\eta\ r_s^n P_n^m(\cos\theta_s)\begin{Bmatrix}\cos m\varphi_s\\ \sin m\varphi_s\end{Bmatrix}\right\} \qquad 3.46$$

where $\eta = \left(2-\delta_m^0\right)\frac{(n-m)!}{(n+m)!}$ $\delta_m^0\begin{cases}=1 & for\ m=0\\ =0 & for\ m\neq 0\end{cases}$

It should be noted that the odd and even terms in the angle brackets are kept separate throughout the following derivation and are recombined at the end. It should also be noted that the summation indices are the same and the summation signs are only repeated for clarity.

3.5.3 The Dipole Source Term

Evaluating the gradient of this term gives:

$$\lambda_{nm} = \eta\begin{Bmatrix}D_x\left(F_r\sin\theta_s\cos\varphi_s + F_\theta\cos\theta_s\cos\varphi_s - F_\varphi\sin\varphi_s\right)+\\ D_y\left(F_r\sin\theta_s\sin\varphi_s + F_\theta\cos\theta_s\sin\varphi_s + F_\varphi\cos\varphi_s\right)+\\ D_z\left(F_r\cos\theta_s + F_\theta\sin\theta_s\right)\end{Bmatrix} \qquad 3.47$$

where:

$$F_r = \sum_{n=1}^{\infty}\sum_{m=0}^{n} n r_s^{n-1} P_n^m(\cos\theta_s)\begin{Bmatrix}\cos m\varphi_s\\ \sin m\varphi_s\end{Bmatrix}$$

$$F_\theta = \sum_{n=1}^{\infty}\sum_{m=0}^{n} r_s^{n-1}\left[-(n+1)P_n^m(\cos\theta_s)\frac{\cos\theta_s}{\sin\theta_s} + \frac{(n-m+1)}{\sin\theta_s}P_{n+1}^m(\cos\theta_s)\right]\begin{Bmatrix}\cos m\varphi_s\\ \sin m\varphi_s\end{Bmatrix} \qquad 3.48$$

$$F_\varphi = \sum_{n=1}^{\infty}\sum_{m=0}^{n} m r_s^{n-1} P_n^m(\cos\theta_s)\frac{1}{\sin\theta_s}\begin{Bmatrix}-\sin m\varphi_s\\ \cos m\varphi_s\end{Bmatrix}$$

3.5.4 The Quadrupole Term

$$\lambda_{nm} = \nabla \cdot \tilde{Q} \cdot \nabla_{r_s} \left\{ \sum_{n=1}^{\infty} \sum_{m=0}^{n} \eta \; r_s^n P_n^m(\cos\theta_s) \begin{Bmatrix} \cos m\varphi_s \\ \sin m\varphi_s \end{Bmatrix} \right\} \qquad 3.49$$

$$\begin{aligned} = \eta \Big\{ & \Big[Q_{xx}\left(F_{xr}\sin\theta_s\cos\varphi_s + F_{x\theta}\cos\theta_s\cos\varphi_s - F_{x\varphi}\sin\varphi_s\right) + \\ & Q_{yx}\left(F_{xr}\sin\theta_s\sin\varphi_s + F_{x\theta}\cos\theta_s\sin\varphi_s + F_{x\varphi}\cos\varphi_s\right) + \\ & Q_{zx}\left(F_{xr}\cos\theta_s + F_{x\theta}\sin\theta_s\right) \Big] \\ & + \\ & Q_{xy}\left(F_{yr}\sin\theta_s\cos\varphi_s + F_{y\theta}\cos\theta_s\cos\varphi_s - F_{y\varphi}\sin\varphi_s\right) + \\ & Q_{yy}\left(F_{yr}\sin\theta_s\sin\varphi_s + F_{y\theta}\cos\theta_s\sin\varphi_s + F_{y\varphi}\cos\varphi_s\right) + \\ & Q_{zy}\left(F_{yr}\cos\theta_s + F_{y\theta}\sin\theta_s\right) \\ & + \\ & \Big[Q_{xz}\left(F_{zr}\sin\theta_s\cos\varphi_s + F_{z\theta}\cos\theta_s\cos\varphi_s - F_{z\varphi}\sin\varphi_s\right) + \\ & Q_{yz}\left(F_{zr}\sin\theta_s\sin\varphi_s + F_{z\theta}\cos\theta_s\sin\varphi_s + F_{z\varphi}\cos\varphi_s\right) + \\ & Q_{zz}\left(F_{zr}\cos\theta_s + F_{z\theta}\sin\theta_s\right) \Big] \Big\} \end{aligned} \qquad 3.50$$

The quadrupole coefficients are merely linear constants . The *F* terms are derived in a two stage process. The first stage is a further vector differentiation from the dipole terms:

$$F_{rr} = \sum_{n=1}^{\infty} \sum_{m=0}^{n} n(n-1) r_s^{n-2} P_n^m(\cos\theta_s) \begin{Bmatrix} \cos m\varphi_s \\ \sin m\varphi_s \end{Bmatrix}$$

$$F_{\theta r} = \sum_{n=1}^{\infty} \sum_{m=0}^{n} (n-1) r_s^{n-2} \left[-(n+1) P_n^m(\cos\theta_s) \frac{\cos\theta_s}{\sin\theta_s} + \frac{(n-m+1)}{\sin\theta_s} P_{n+1}^m(\cos\theta_s) \right] \begin{Bmatrix} \cos m\varphi_s \\ \sin m\varphi_s \end{Bmatrix}$$

$$F_{\varphi r} = \sum_{n=1}^{\infty} \sum_{m=0}^{n} m(n-1) r_s^{n-2} P_n^m(\cos\theta_s) \frac{1}{\sin\theta_s} \begin{Bmatrix} -\sin m\varphi_s \\ \cos m\varphi_s \end{Bmatrix} \qquad 3.51$$

$$F_{r\theta} = \sum_{n=1}^{\infty} \sum_{m=0}^{n} n r_s^{n-2} \left[-(n+1) P_n^m(\cos\theta_s) \frac{\cos\theta_s}{\sin\theta_s} + \frac{(n-m+1)}{\sin\theta_s} P_{n+1}^m(\cos\theta_s) \right] \begin{Bmatrix} \cos m\varphi_s \\ \sin m\varphi_s \end{Bmatrix}$$

$$F_{\theta\theta} = \sum_{n=1}^{\infty} \sum_{m=0}^{n} r_s^{n-2} \begin{bmatrix} -(n+1) \left\{ P_n^m(\cos\theta_s) \frac{\cos\theta_s}{\sin\theta_s} - P_n^m(\cos\theta_s) - P_n^m(\cos\theta_s) \frac{\cos^2\theta_s}{\sin^2\theta_s} \right\} + \\ (n-m+1) \left\{ \frac{1}{\sin\theta_s} P_{n+1}^m(\cos\theta_s) - \frac{\cos\theta_s}{\sin^2\theta_s} P_{n+1}^m(\cos\theta_s) \right\} \end{bmatrix} \begin{Bmatrix} \cos m\varphi_s \\ \sin m\varphi_s \end{Bmatrix} \qquad 3.52$$

$$F_{\varphi\theta} = \sum_{n=1}^{\infty}\sum_{m=0}^{n} m r_s^{n-2}\left\{P_n^{m}(\cos\theta_s)\frac{1}{\sin\theta_s} - P_n^{m}(\cos\theta_s)\frac{\cos\theta_s}{\sin^2\theta_s}\right\}\begin{Bmatrix}-\sin m\varphi_s\\ \cos m\varphi_s\end{Bmatrix} \qquad 3.52(\text{cont})$$

$$F_{r\varphi} = \sum_{n=1}^{\infty}\sum_{m=0}^{n} nm\frac{r_s^{n-2}}{\sin\theta_s}P_n^{m}(\cos\theta_s)\begin{Bmatrix}-\sin m\varphi_s\\ \cos m\varphi_s\end{Bmatrix}$$

$$F_{\theta\varphi} = \sum_{n=1}^{\infty}\sum_{m=0}^{n} m\frac{r_s^{n-2}}{\sin\theta_s}\left[-(n+1)P_n^{m}(\cos\theta_s)\frac{\cos\theta_s}{\sin\theta_s} + \frac{(n-m+1)}{\sin\theta_s}P_{n+1}^{m}(\cos\theta_s)\right]\begin{Bmatrix}-\sin m\varphi_s\\ \cos m\varphi_s\end{Bmatrix} \qquad 3.53$$

$$F_{\varphi\varphi} = \sum_{n=1}^{\infty}\sum_{m=0}^{n} -m^2 r_s^{n-2}P_n^{m}(\cos\theta_s)\frac{1}{\sin^2\theta_s}\begin{Bmatrix}\cos m\varphi_s\\ \sin m\varphi_s\end{Bmatrix}$$

where $P'^{m}_{n}(\cos\theta_s)$ is the first differential with respect to θ_s of the Associated Legendre term. These terms are combined as follows:

$$F_{xr} = F_{rr}\sin\theta_s\cos\varphi_s + F_{\theta r}\cos\theta_s\cos\varphi_s - F_{\varphi r}\sin\varphi_s \qquad 3.54$$

$$F_{yr} = F_{rr}\sin\theta_s\sin\varphi_s + F_{\theta r}\cos\theta_s\sin\varphi_s + F_{\varphi r}\cos\varphi_s$$

$$F_{zr} = F_{rr}\cos\theta_s - F_{\theta r}\sin\theta_s$$

$$F_{x\theta} = F_{r\theta}\sin\theta_s\cos\varphi_s + F_{\theta\theta}\cos\theta_s\cos\varphi_s - F_{\varphi\theta}\sin\varphi_s + F_r\frac{\cos\theta_s\cos\varphi_s}{r_s} - F_\theta\frac{\sin\theta_s\cos\varphi_s}{r_s}$$

$$F_{y\theta} = F_{r\theta}\sin\theta_s\sin\varphi_s + F_{\theta\theta}\cos\theta_s\sin\varphi_s + F_{\varphi\theta}\cos\varphi_s + F_r\frac{\cos\theta_s\sin\varphi_s}{r_s} - F_\theta\frac{\sin\theta_s\sin\varphi_s}{r_s}$$

$$F_{z\theta} = F_{r\theta}\cos\theta_s - F_{\theta\theta}\sin\theta_s - F_\theta\frac{\cos\theta_s}{r_s} - F_r\frac{\sin\theta_s}{r_s}$$

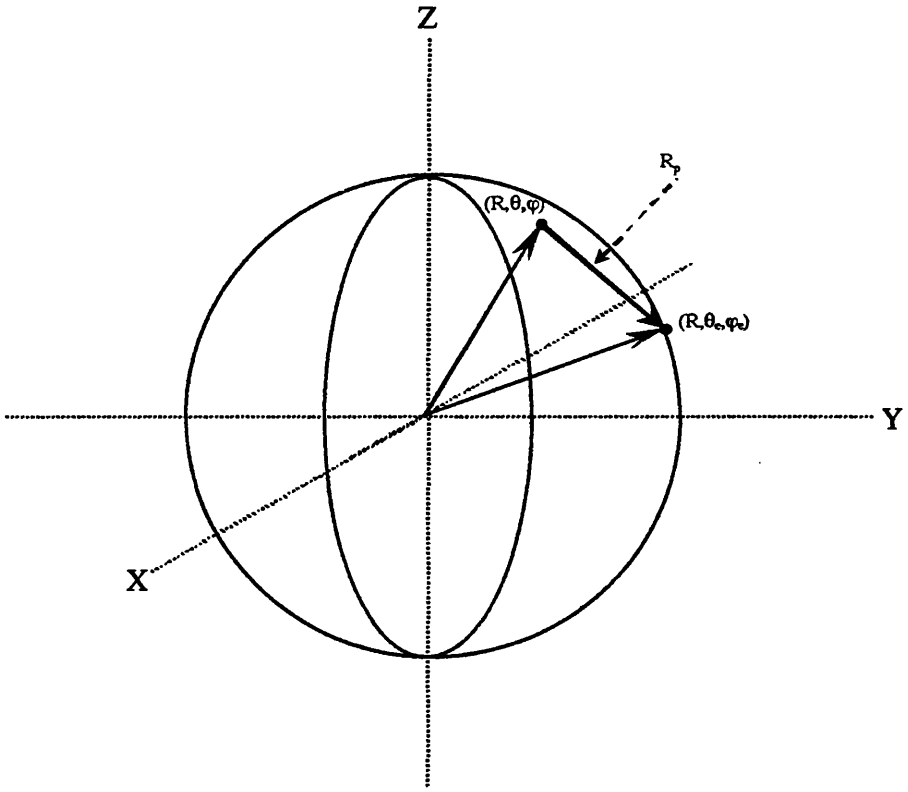
$$F_{x\varphi} = F_{r\varphi}\sin\theta_s\cos\varphi_s + F_{\theta\varphi}\cos\theta_s\cos\varphi_s - F_{\varphi\varphi}\sin\varphi_s - F_r\frac{\sin\varphi_s}{r_s} - F_\theta\frac{\cos\theta_s\sin\varphi_s}{r_s\sin\theta_s} - F_\varphi\frac{\cos\varphi_s}{r_s\sin\theta_s}$$

$$F_{y\varphi} = F_{r\varphi}\sin\theta_s\sin\varphi_s + F_{\theta\varphi}\cos\theta_s\sin\varphi_s + F_{\varphi\varphi}\cos\varphi_s - F_r\frac{\cos\varphi_s}{r_s} + F_\theta\frac{\cos\theta_s\cos\varphi_s}{r_s\sin\theta_s} - F_\varphi\frac{\sin\varphi_s}{r_s\sin\theta_s}$$

$$F_{z\varphi} = F_{r\varphi}\cos\theta_s - F_{\theta\varphi}\sin\theta_s$$

3.55

3.5.5 The Surface Integration



The magnetic vector potential generated by the volume currents is:

$$\bar{A}(\bar{r}) = \frac{\mu}{4\pi}\oint_S \frac{\sigma V}{R_p}\,\bar{dS} \qquad 3.56$$

The reciprocal of the space vector pointing from any surface point $(R,\theta,\varphi)$to the point of evaluation (R,θ_e,φ_e)can be expanded as follows:

$$\frac{1}{R}\sum_{n'}^{\infty}\sum_{m'=0}^{n'}\eta' \cos\left(m'[\varphi-\varphi_e]\right)P_{n'}^{m'}(\cos\theta)P_{n'}^{m'}(\cos\theta_e) \qquad 3.57$$

where primes are used to distinguish these summations from those of the source multipole series above. The vector area $\bar{ds}$ is expanded in spherical co-ordinates with rectangular unit vectors:

$$\bar{ds} = \left(\hat{u}_x \sin\theta\cos\varphi + \hat{u}_y \sin\theta\sin\varphi + \hat{u}_z\cos\theta\right)R^2\sin\theta\; d\theta\; d\varphi \qquad 3.58$$

Then substituting the expression for V above the integral becomes:

$$\bar{A}(\bar{r}) = \frac{\mu\sigma}{4\pi}\int_{\varphi=0}^{2\pi}\int_{\theta=0}^{\pi}\left[\frac{1}{4\pi\sigma}\sum_{n=1}^{\infty}\sum_{m=0}^{n}\frac{2n+1}{n}\frac{\lambda_{nm}}{R^{n+1}}P_n^m(\cos\theta)\begin{Bmatrix}\cos m\varphi\\ \sin m\varphi\end{Bmatrix}\right]\cdot$$
$$\left[\frac{1}{R}\sum_{n'}^{\infty}\sum_{m'=0}^{n'}\eta'\cos\left(m'[\varphi-\varphi_e]\right)P_{n'}^{m'}(\cos\theta)P_{n'}^{m'}(\cos\theta_e)\right]\cdot \qquad 3.59$$
$$\left(\hat{u}_x\sin\theta\cos\varphi + \hat{u}_y\sin\theta\sin\varphi + \hat{u}_z\cos\theta\right)R^2\sin\theta\ d\theta\ d\varphi$$

Due to the orthogonality properties of the trigonometric functions, only certain combinations of m, n, m' and n' are non-zero. The integration is carried out in Appendix 1 with the following result:

- The $(\hat{u}_x)$ terms

$$\sum_{n=1}^{\infty} f_x\begin{Bmatrix}A\\ B\end{Bmatrix} = \sum_{n=1}^{\infty}\frac{1}{2(2n+1)}\cdot$$
$$\left(\sum_{m=0}^{n}\lambda_{nm}\frac{(n+m)(n+m-1)}{2n-1}P_{n-1}^{m-1}(\cos\theta_e) - \sum_{m=1}^{n+2}\lambda_{nm}\frac{(n+m+2)(n+m+1)}{2n+3}P_{n+1}^{m-1}(\cos\theta_e)\right)\begin{Bmatrix}\cos(m-1)\varphi_e\\ \sin(m-1)\varphi_e\end{Bmatrix}$$
$$+$$
$$\sum_{n=1}^{\infty}\frac{1}{(2n+1)}\cdot$$
$$\left(\sum_{m=0}^{n}\frac{\lambda_{nm}}{(2-\delta_m^0)}\frac{P_{n+1}^{m+1}(\cos\theta_e)}{2n+3} - \sum_{m=0}^{n-2}\frac{\lambda_{nm}}{(2-\delta_m^0)}\frac{P_{n-1}^{m+1}(\cos\theta_e)}{2n-1}\right)\begin{Bmatrix}\cos(m+1)\varphi_e\\ (1-\delta_m^0)\sin(m+1)\varphi_e\end{Bmatrix}$$

3.60

- The $(\hat{u}_y)$ terms

$$\sum_{n=1}^{\infty} f_y \begin{Bmatrix} A \\ B \end{Bmatrix} = \sum_{n=1}^{\infty} \frac{1}{2(2n+1)} \cdot \begin{pmatrix} \sum_{m=0}^{n} \lambda_{nm} \frac{(n+m)(n+m-1)}{2n-1} P_{n-1}^{m-1}(\cos\theta_e) - \\ \sum_{m=1}^{n+2} \lambda_{nm} \frac{(n+m+2)(n+m+1)}{2n+3} P_{n+1}^{m-1}(\cos\theta_e) \end{pmatrix} \begin{Bmatrix} \cos(m-1)\varphi_e \\ -\sin(m-1)\varphi_e \end{Bmatrix}$$

$$+$$

$$\sum_{n=1}^{\infty} \frac{1}{(2n+1)} \cdot \begin{pmatrix} \sum_{m=0}^{n} \frac{\lambda_{nm}}{(2-\delta_m^0)} \frac{P_{n+1}^{m+1}(\cos\theta_e)}{2n+3} - \\ \sum_{m=0}^{n-2} \frac{\lambda_{nm}}{(2-\delta_m^0)} \frac{P_{n-1}^{m+1}(\cos\theta_e)}{2n-1} \end{pmatrix} \begin{Bmatrix} \cos(m+1)\varphi_e \\ (1-\delta_m^0)\sin(m+1)\varphi_e \end{Bmatrix}$$

3.61

- The $(\hat{u}_z)$ term

$$\sum_{n=1}^{\infty} f_z \begin{Bmatrix} A \\ B \end{Bmatrix} = \sum_{n=1}^{\infty} \sum_{m=0}^{n} \lambda_{nm} \frac{1}{(2n+1)} \cdot \begin{pmatrix} \frac{(n-m+1)}{2n+3} P_{n+1}^{m}(\cos\theta_e) + \\ \frac{(n+m)}{2n-1} P_{n-1}^{m}(\cos\theta_e) \end{pmatrix} \begin{Bmatrix} \cos m\varphi_e \\ \sin m\varphi_e \end{Bmatrix} \qquad 3.62$$

3.5.6 The Magnetic Vector Potential Curl

The final magnetic field correction due to the volume currents is the result of taking the curl of the magnetic vector potential:

$$\bar{B}(\bar{r}) = \nabla_e \times \bar{A}(\bar{r}) = \frac{\mu}{4\pi} \nabla_e \times \oint_S \frac{\sigma V}{R_p} \overline{dS} \qquad 3.63$$

In terms of the integration products from above:

$$\bar{B}(\bar{r}) = \frac{\mu}{4\pi} \nabla_e \times \left(\sum_{n=1}^{\infty} \frac{2n+1}{n} \frac{1}{R^{n+1}} \frac{1}{R} R^2 \{f_x, f_y, f_z\} \right) \qquad 3.64$$

The curl derivation is easiest to carry out in spherical co-ordinates so the following notation will be used:

$$\begin{aligned} f_R &= \frac{1}{R^n}\left(f_x \sin\theta_e \cos\varphi_e + f_y \sin\theta_e \sin\varphi_e + f_z \cos\theta_e\right) \\ f_\theta &= \frac{1}{R^n}\left(f_x \cos\theta_e \cos\varphi_e + f_y \cos\theta_e \sin\varphi_e - f_z \sin\theta_e\right) \\ f_\varphi &= \frac{1}{R^n}\left(-f_x \sin\varphi_e + f_y \cos\varphi_e\right) \end{aligned} \qquad 3.65$$

and $$f'^{\beta}_{\alpha} = \frac{\partial f_\alpha}{\partial \beta}$$

The following shows only the active elements of the curl derivation:

$$\frac{1}{R^2 \sin\theta_e} \begin{vmatrix} \hat{u}_R & R_e \hat{u}_{\theta_e} & R_e \sin\theta_e \hat{u}_{\varphi_e} \\ \partial/\partial R & \partial/\partial\theta_e & \partial/\partial\varphi_e \\ f_R & R f_\theta & R \sin\theta_e f_\varphi \end{vmatrix} \qquad 3.66$$

and defining:

$$\begin{aligned} f'^{\varphi}_{R} &= f'^{\varphi}_{x} \sin\theta_e \cos\varphi_e + f'^{\varphi}_{y} \sin\theta_e \sin\varphi_e + f'^{\varphi}_{z} \cos\theta_e - \\ &\quad f_x \sin\theta_e \sin\varphi_e + f_y \sin\theta_e \cos\varphi_e \\ f'^{\theta}_{R} &= f'^{\theta}_{x} \sin\theta_e \cos\varphi_e + f'^{\theta}_{y} \sin\theta_e \sin\varphi_e + f'^{\theta}_{z} \cos\theta_e + \\ &\quad f_x \cos\theta_e \cos\varphi_e + f_y \cos\theta_e \sin\varphi_e - f_z \sin\theta_e \\ f'^{\varphi}_{\theta} &= f'^{\varphi}_{x} \cos\theta_e \cos\varphi_e + f'^{\varphi}_{y} \cos\theta_e \sin\varphi_e - f'^{\varphi}_{z} \sin\theta_e - \\ &\quad f_x \cos\theta_e \sin\varphi_e + f_y \cos\theta_e \cos\varphi_e \\ f'^{\theta}_{\varphi} &= -f'^{\theta}_{x} \sin\varphi_e + f'^{\theta}_{y} \cos\varphi_e \end{aligned} \qquad 3.67$$

The differentials $f_x'^{\theta}, f_y'^{\theta}, f_z'^{\theta}, f_x'^{\varphi}, f_y'^{\varphi}, f_z'^{\varphi}$ are straight forward and are not detailed here but can be seen in the model coding Appendix 8.

The curl operation results in the (R,θ,φ) components which are given shorthand forms $\left(f_{\hat{u}_R}, f_{\hat{u}_\theta}, f_{\hat{u}_\varphi}\right)$. Setting $R = 1$,which will be the case evaluated, gives:

$$\begin{aligned} f_{\hat{u}_R} &= \frac{1}{\sin\theta_e}\left(\cos\theta_e\ f_\varphi + \sin\theta_e f_\varphi'^{\theta}\right) - \frac{1}{\sin\theta_e} f_\theta'^{\varphi} \\ f_{\hat{u}_\theta} &= \frac{1}{\sin\theta_e} f_R'^{\varphi} - (n-1) f_\varphi \\ f_{\hat{u}_\varphi} &= -f_R'^{\theta} - (n-1) f_\theta \end{aligned} \qquad 3.68$$

Then the final form in cartesian co-ordinates is:

$$\begin{aligned} B_{vx} &= \frac{\mu}{4\pi}\left\{f_{\hat{u}_R}\sin\theta_e\cos\varphi_e + f_{\hat{u}_\theta}\cos\theta_e\cos\varphi_e - f_{\hat{u}_\varphi}\sin\varphi_e\right\} \\ B_{vy} &= \frac{\mu}{4\pi}\left\{f_{\hat{u}_R}\sin\theta_e\sin\varphi_e + f_{\hat{u}_\theta}\cos\theta_e\sin\varphi_e + f_{\hat{u}_\varphi}\cos\varphi_e\right\} \\ B_{vz} &= \frac{\mu}{4\pi}\left\{f_{\hat{u}_R}\cos\theta_e - f_{\hat{u}_\theta}\sin\theta_e\right\} \end{aligned} \qquad 3.69$$

3.6 Conclusion

The equations developed above are for the electric potential and magnetic field on the surface of a homogeneous sphere due to an internal arbitrarily located point multipole current source. The electric potential case for the dipole source is from a closed form lead field formula published over 25 years ago and this has been extended to cover the quadrupole current case and the infinite medium magnetic field for both cases. The advantage of a closed form expression is that infinite series are avoided which leads to more efficient modelling. However, the magnetic field requires adjustment due to the behaviour of the Coulombian currents at the sphere's surface boundary and this involves a surface integration which has no closed form equivalent.

The surface integration developed in this section is lengthy and seemingly leads to a fairly clumsy expression. However, the parts of the expression relating to the point source co-ordinates and the field co-ordinates can be separated. Only a single computation due to the latter is required once the field measurement co-ordinates are

fixed and the results, although numerous, are then treated in the inverse model as constants.

The expressions contain a number of Associated Legendre Polynomials and these could be rationalised (and indeed have been) using recursion relationships. However, the computational efficiency gained is outweighed by the loss of clarity in the model coding so that they are left mostly in the form they appear above. The only exceptions are expressions, such as the differentials of expressions with Associated Legendres, where the denominator contains a $\sin \theta_e$. A field at, or close to, the one of the poles could not therefore be computed so in the model alternative expressions are developed which avoid this.

The technique developed above can be extended to any higher order point current multipole source by successive differentiations although the number of terms will increase by the square. In the least squares inverse solution a further differentiation of the quadrupole expressions is required in order to solve the minimisation problem. This differentiation sets the direction for the next guess of the quadrupole location which therefore effectively minimises the octapole term.

4. Development of Generalised Multipole Equations for the Bounded Prolate Spheroid

The prolate spheroid was examined since a portion of its surface could provide a better fit to the surface of the thorax. The purpose of this section is to show that the same method as for the sphere in the previous section can be applied and can yield very similar equations. These equations have not been computer modelled in this Thesis.

Very little work has been done on the spheroid but one paper [Cuffin and Cohen:1977] examined a restricted dipole case. The equations developed here are more general but a reconciliation with this paper is carried out in Appendix 3.5. As with the criteria for the sphere, the results are expressed in Cartesian co-ordinates and the point source location is arbitrary. [Fiesler et al:1995] compared the sphere and spheroid using this Cuffin and Cohen analysis.

There are some differences in co-ordinate systems in the literature. This analysis uses a right handed co-ordinate system as defined in Appendix 3.1. The arbitrarily located source is at $(\eta_0, \xi_0, \varphi_0)$ and the measurements are on a constant spheroidal surface (η_e, ξ, φ).

4.1 Prolate Spheroid - The Electric Potential from a Dipole

The potential on the surface of a bounded prolate spheroid is derived by [Yeh and Martinek:1959] from a consideration of the solution to the Laplace equation for the case of zero normal derivative of the potential at the surface boundary. However, this solution contains Associated Legendre Polynomials of the second kind. A paper by [Berry:1956] on the oblate spheroid uses the Wronksian relationship to replace these with a term containing the derivative of the Legendre Polynomials of the first kind. Making this substitution in the prolate spheroid yields an expression for the surface potential:

$$\Phi_{(\eta,\xi)} = -\frac{1}{4\pi\sigma c}\sum_{n=0}^{\infty}\sum_{m=0}^{n}\left(2-\delta_m^0\right)(2n+1)\frac{(n-m)!}{(n+m)!}\frac{1}{\eta_e^2-1}\cdot \frac{P_n^m(\xi)}{P_n'^m(\eta_e)}\overline{D}\cdot\nabla\left[P_n^m(\eta_0)P_n^m(\xi_0)\cos m(\varphi-\varphi_0)\right] \qquad 4.1$$

where $2c$ is the interfocal distance.

The expansion of the gradient of the space vector which points from the source to the surface leads to the equivalent multipole expansion at the co-ordinate origin (the centre of the spheroid) for the arbitrarily located point dipole source.

This equation contains the infinite medium potential at the surface position which may be written more conventionally in terms of multipole components and then compared with the electric potential case for the sphere:

$$\Phi_{(\eta,\xi)} = \frac{1}{4\pi\sigma}\sum_{n=0}^{\infty}\sum_{m=0}^{n}\left(A_{nm}\cos m\varphi + B_{nm}\sin m\varphi\right)P_n^m(\eta_e)P_n^m(\xi) \qquad 4.2$$

where:

$$\begin{Bmatrix}A_{nm}\\B_{nm}\end{Bmatrix} = \frac{1}{c}\left(2-\delta_m^0\right)(-1)^m(2n+1)\left[\frac{(n-m)!}{(n+m)!}\right]^2 \cdot \overline{D}\cdot\nabla\left[P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right] \qquad 4.3$$

These components are evaluated in cartesian form in Appendix 3.2 for the dipole and quadrupole terms with the following result:

$$A_{10} = \frac{3\langle D_z\rangle}{c^2} \qquad A_{11} = -\frac{3\langle D_x\rangle}{2c^2}j \qquad B_{11} = -\frac{3\langle D_y\rangle}{2c^2}j$$

$$A_{20} = 5\frac{3}{2c^2}\langle 2z_0D_z - x_0D_x - y_0D_y\rangle$$

$$A_{21} = -\frac{5}{2c^2}\langle z_0D_x + x_0D_z\rangle j \qquad B_{21} = -\frac{5}{2c^2}\langle z_0D_y + y_0D_z\rangle j \qquad 4.4$$

$$A_{22} = -\frac{1}{8c^2}\frac{5}{2}\langle x_0D_x - y_0D_y\rangle \qquad B_{22} = -\frac{1}{8c^2}\frac{5}{2}\langle y_0D_x + x_0D_y\rangle$$

where $j^2 = -1$

Terms in the brackets are the same multipole components as the sphere.

This form of equation for the infinite medium may now be incorporated into the expression for the bounded spheroid:

$$\Phi_{(\eta,\xi)} = -\frac{1}{4\pi\sigma}\sum_{n=0}^{\infty}\sum_{m=0}^{n}\alpha_{nm}P_n^m(\xi)\left[A_{nm}\cos m\varphi + B_{nm}\sin m\varphi\right] \qquad 4.5$$

$$\text{where} \quad \alpha_{nm} = (-1)^m \frac{(n+m)!}{(n-m)!} \frac{1}{\eta_e^2 - 1} \frac{1}{P_n'^m(\eta_e)} \qquad 4.6$$

Note that the imaginary components for the *m (odd)* series will become real as a consequence of $\eta \geq 1$ in the *m (odd)* Legendre Polynomials.

4.2 Prolate Spheroid - The Electric Potential from a Quadrupole

In the same way that the quadrupole was derived for the sphere, the electric potential due to a quadrupole source is derived from a further differentiation of the source term. Using the infinite medium expression:

$$\Phi_{(\eta,\xi)} = \frac{1}{4\pi\sigma} \sum_{n=0}^{\infty} \sum_{m=0}^{n} (\mathcal{A}_{nm} \cos m\varphi + \mathcal{B}_{nm} \sin m\varphi) P_n^m(\eta_e) P_n^m(\xi) \qquad 4.7$$

where:

$$\begin{Bmatrix} \mathcal{A}_{nm} \\ \mathcal{B}_{nm} \end{Bmatrix} = \frac{1}{c}(2 - \delta_m^0)(-1)^m (2n+1) \left[\frac{(n-m)!}{(n+m)!}\right]^2 \cdot \left\{ \nabla \cdot \tilde{Q} \cdot \nabla \left[P_n^m(\eta_0) P_n^m(\xi_0) \begin{Bmatrix} \cos m\varphi_0 \\ \sin m\varphi_0 \end{Bmatrix} \right] \right\} \qquad 4.8$$

The first term of this multipole expansion is evaluated in Appendix 3.2a

$$\mathcal{A}_{20} = \frac{15}{2c^3}\langle 2Q_{zz} - Q_{xx} - Q_{yy} \rangle$$

$$\mathcal{A}_{21} = -\frac{5}{2c^3}\langle Q_{xz} + Q_{zx} \rangle j \qquad \mathcal{B}_{21} = -\frac{5}{2c^3}\langle Q_{zy} + Q_{yz} \rangle j \qquad 4.9$$

$$\mathcal{A}_{22} = -\frac{1}{8c^3}\frac{5}{2}\langle Q_{xx} - Q_{yy} \rangle \qquad \mathcal{B}_{22} = -\frac{1}{8c^3}\frac{5}{2}\langle Q_{xy} + Q_{yx} \rangle$$

The derivations for further terms are extremely long and not carried out here since a model to evaluate the series would be coded in terms of the quadrupole tensor as for the sphere. As with the dipole series, these terms can be inserted into the bounded prolate spheroid equation:

$$\Phi_{(\eta,\xi)} = -\frac{1}{4\pi\sigma} \sum_{n=0}^{\infty} \sum_{m=0}^{n} \alpha_{nm} P_n^m(\xi) [\mathcal{A}_{nm} \cos m\varphi + \mathcal{B}_{nm} \sin m\varphi] \qquad 4.10$$

4.3 Prolate Spheroid - The Magnetic Field

As with the sphere, the magnetic field is derived via the magnetic vector potential in two parts; the direct field from the source and the correction due to the volume currents at the boundary:

$$\overline{A}(\eta_e,\xi) = \frac{\mu}{4\pi}\int_v \frac{\overline{J}^*}{R}\,dv - \frac{\mu}{4\pi}\oint_s \frac{\Phi_{(\eta_e,\xi)}}{R}\,\overline{ds} \qquad 4.11$$

$$\overline{B}(\eta_e,\xi) = \nabla \times \overline{A}(\eta_e,\xi) \qquad 4.12$$

The direct field is computed more straightforwardly using:

for the dipole: $$\overline{B_d} = -\frac{\mu}{4\pi}\left[\overline{D}_{(x,y,z)} \times \nabla\left(\frac{1}{R}\right)\right] \qquad 4.13$$

for the quadrupole: $$\overline{B_d} = \frac{\mu}{4\pi}\left[\left(\tilde{Q}\cdot\nabla\right) \times \nabla\left(\frac{1}{R}\right)\right] \qquad 4.14$$

where

$$\nabla\left(\frac{1}{R}\right) = \left(2-\delta_m^0\right)(-1)^m(2n+1)\left[\frac{(n-m)!}{(n+m)!}\right]^2 \cdot P_n^m(\eta_e)P_n^m(\xi)\nabla\left\{Q_n^m(\eta_0)P_n^m(\xi_0)\cos m(\varphi-\varphi_0)\right\} \qquad 4.15$$

Note that the Q in the above is an Associated Legendre Polynomial of the second kind. The last part of this expression is a differentiation in prolate spheroid co-ordinates followed by a conversion to cartesian co-ordinates:

$$\nabla\left\{Q_n^m(\eta_0)P_n^m(\xi_0)\cos m(\varphi-\varphi_0)\right\} = \left[\frac{1}{h_\eta}\frac{c\eta}{h_\xi}\Gamma_\eta\cos\varphi - \frac{1}{h_\xi}\frac{c\xi}{h_\eta}\Gamma_\xi\cos\varphi - \frac{1}{h_\varphi}\Gamma_\varphi\sin\varphi\right]\hat{i} \left[\frac{1}{h_\eta}\frac{c\eta}{h_\xi}\Gamma_\eta\sin\varphi - \frac{1}{h_\xi}\frac{c\xi}{h_\eta}\Gamma_\xi\sin\varphi + \frac{1}{h_\varphi}\Gamma_\varphi\cos\varphi\right]\hat{j} \left[\frac{1}{h_\eta}\frac{c\xi}{h_\eta}\Gamma_\eta + \frac{1}{h_\xi}\frac{c\eta}{h_\xi}\Gamma_\xi\right]\hat{k} \qquad 4.16$$

where

$$\begin{aligned}\Gamma_\eta &= Q_n'^m(\eta_0)P_n^m(\xi_0)\cos m(\varphi-\varphi_0)\\ \Gamma_\xi &= Q_n'^m(\eta_0)P_n'^m(\xi_0)\cos m(\varphi-\varphi_0)\\ \Gamma_\varphi &= -Q_n'^m(\eta_0)P_n^m(\xi_0)m\sin m(\varphi-\varphi_0)\end{aligned} \qquad 4.17$$

the primes indicating the first differential.

4.4 Prolate Spheroid - The Influence of the Surface Boundary

The surface integration required for prolate spheroid is approached in a slightly different way from the sphere. Firstly, the integration required is:

$$\overline{A}_{v(\eta_e,\xi)} = -\frac{\mu}{4\pi}\oint_s \frac{\Phi_{(\eta,\xi)}}{R}\overline{ds} \qquad 4.18$$

where:

$\Phi_{(\eta,\xi)}$ is as above

$\overline{ds} = \hat{\eta}h_\xi h_\varphi\, d\xi\, d\varphi$ $\quad\hat{\eta}$ is the unit vector in the η direction

$$\frac{1}{R} = \frac{1}{c}\sum_{n'=0}^{\infty}\sum_{m'=0}^{n'}\left(2-\delta_{m'}^0\right)(2n'+1)\left[\frac{(n'-m')!}{(n'+m')!}\right]^2 \cdot Q_{n'}^{m'}(\eta_e)P_{n'}^{m'}(\xi_e)P_{n'}^{m'}(\eta_e)P_{n'}^{m'}(\xi)\cos m'(\varphi_e-\varphi) \qquad 4.19$$

where the field point is (η_e,ξ_e,φ_e) and the source point is any point on the surface (η_e,ξ,φ)

The integration limits are $-1 \le \xi \le 1$ and $0 \le \varphi \le 2\pi$

Note that the summation indices are primed to distinguish them from the summation indices of the surface potential.

The integration is more easily carried out if:

$$\nabla \times \overline{A}_{v(\eta_e,\xi_e)} \text{ is expressed as } -\frac{\mu}{4\pi}\oint_s \Phi_{(\eta,\xi)}\,\nabla\left(\frac{1}{R}\right)\times\overline{ds} \qquad 4.20$$

where

$$\nabla\left(\frac{1}{R}\right)\times\overline{ds} = \left[\begin{array}{c}\left\{-c\xi\frac{h_\xi}{h_\eta}\frac{\partial}{\partial\varphi}\left(\frac{1}{R}\right)\begin{pmatrix}\cos\varphi\\ \sin\varphi\end{pmatrix} - h_\varphi\frac{\partial}{\partial\xi}\left(\frac{1}{R}\right)\begin{pmatrix}-\sin\varphi\\ \cos\varphi\end{pmatrix}\right\}\begin{pmatrix}\hat{i}\\ \hat{j}\end{pmatrix}\\ +\\ c\eta\frac{\partial}{\partial\varphi}\left(\frac{1}{R}\right)\hat{k}\end{array}\right] d\varphi\, d\xi$$

4.21

The integrals are lengthy and are carried out for the dipole series in Appendices 3.3 and 3.4 with the following result:

$$\bar{B}_{v(\eta_e,\xi)} = -\frac{\mu}{4\pi}\frac{1}{\left(\eta_e^2-1\right)^{1/2}}\sum_{n=1}^{\infty}\beta_{n0}^{*}A_{n0}\begin{bmatrix}\sin\varphi_e & \hat{i}\\ \cos\varphi_e & \hat{j}\end{bmatrix} +$$

$$\sum_{m=1}^{n}\begin{Bmatrix}A_{nm}\\ B_{nm}\end{Bmatrix}\begin{bmatrix} \frac{1}{2}\begin{Bmatrix}-\sin(m+1)\varphi_e\beta_{nm}^{*}-\sin(m-1)\varphi_e\beta_{nm}^{**}\\ \cos(m+1)\varphi_e\beta_{nm}^{*}+\cos(m-1)\varphi_e\beta_{nm}^{**}\end{Bmatrix}\hat{i} \\ \frac{1}{2}\begin{Bmatrix}\cos(m+1)\varphi_e\beta_{nm}^{*}-\cos(m-1)\varphi_e\beta_{nm}^{**}\\ \sin(m+1)\varphi_e\beta_{nm}^{*}-\sin(m-1)\varphi_e\beta_{nm}^{**}\end{Bmatrix}\hat{j} \\ \left(-\frac{m\eta_e Q_n^m(\eta_e)P_n^m(\xi_e)P_n^m(\eta_e)}{\left(\eta_e^2-1\right)^{1/2}P_n'^m(\eta_e)}\right)\begin{Bmatrix}\sin m\varphi_e\\ -\cos m\varphi_e\end{Bmatrix}\hat{k}\end{bmatrix} \qquad 4.22$$

where

$$\beta_{nm}^{*} = \frac{Q_n^{m+1}(\eta_e)P_n^{m+1}(\xi_e)P_n^{m+1}(\eta_e)}{(n+m+1)(n-m)P_n'^m(\eta_e)}$$

$$\beta_{nm}^{**} = \frac{Q_n^{m-1}(\eta_e)P_n^{m-1}(\xi_e)P_n^{m-1}(\eta_e)}{P_n'^m(\eta_e)}\left[(n-m+1)(n+m)\right]^2 \qquad 4.23$$

Note that for β_{nm}^{*} the case of n=m does not exist.

4.5 Prolate Spheroid – Conclusion

The above derivation is lengthy and as a check a reconciliation has been carried out with the most relevant published work [Cuffin and Cohen:1977]. These authors analyse the prolate spheroid from a more restricted perspective since they evaluate the source confined to the $\varphi_0 = 0$ axis. Thus there are no B_{nm} multipole components. They also use a left handed co-ordinate system so the $\hat{i}$ and $\hat{j}$ components are reversed. The reconciliation with this thesis is carried out in Appendix 3.5.

The general formulae developed in this section for the electric potential and the magnetic field on the surface of a bounded prolate spheroid from an arbitrarily located multipole current source, have not yet appeared in the literature and are suitable for computer modelling.

5. Non-linear Least Squares Modelling

All the forward analytical models developed in this thesis have the same general format. Each of m multipole components, a_j (j=1 ...m), in the source multiplies the summation of a number of closed or infinite series position functions Φ_j. The position functions are expressions containing only the i sets of surface co-ordinates t_i and source co-ordinates α. The t_i surface co-ordinates are the positions of the i(1...n) field sensing elements and are therefore fixed.

In this section, lower case bold letters will represent column vectors while normal typeface letters with a subscript will mean a component of that vector. Upper case bold letters will represent matrices. Superscript symbol j means iteration step j and other superscripts are common ones used in linear algebra eg T=transpose, -1= inverse, $\perp$=orthogonal.

In the inverse model the co-ordinates of the one or more point sources are the objective to be determined from the minimization of some merit function which evaluates as a single number the difference between the forward calculated field and the measured field for all *n* sensing locations. The forward model computes the electric potential and/or each orthogonal plane of the magnetic field at the n field points and for any point i:

$$v_i = \sum_{j=1}^{m} a_j \Phi_j(\alpha, t_i) \qquad 5.1$$

The a_j are linear variables and many methods have been developed for linear least squares solutions. These methods have at their heart a technique for assessing the direction of the next guess for the linear variables and moving along that direction either as far as possible or some pre-determined distance. The problem with the α non-linear unknowns is not only to find the direction in which to move but to move an optimal distance, which due to the non-linearity is not necessarily the maximum distance. One advantage of the analytical equations developed in this Thesis is that, although they are lengthy, they are continuous and differentiable which suggests that efficient least squares solutions should be applicable since both the gradient and more importantly the rate of gradient change can be computed.

The non-linear least squares technique favoured by most authors in the ECG and MCG inverse modelling arena is the Levenburg-Marquardt method which was proposed by [Marquardt:1963]. This method is well documented in the literature [eg Vetterling et al:1985,Demas and Demas 1989] and is only described briefly here.

The disadvantage of the Levenburg-Marquardt method is firstly that linear and non-linear unknowns are given equal status and, since the non-linear unknowns require greater computation time, it is likely to be inefficient. Secondly, the method requires the inversion of an n x n matrix for each field measurement set. This is computationally intensive with the added risk of singularities causing complete inversion failure.

Methods for separating linear from non-linear unknowns are not new but have only been applied in this field relatively recently [Oostendorp and van Oosterom:1993]. The Levenburg-Marquardt approach in this thesis is to make several iterations using the non-linear variables, until convergence slows to some pre-determined point, and then a single iteration to find a new set of linear parameters, repeating the whole cycle until a satisfactory least squares error is reached. However, this is not true separation of variables since the minimisation method is the same for each type of variable.

The variable projection method was first described by [Golub and Pereyra:1973]. [Oostendorp and van Oosterom:1993] had applied it to a relatively straightforward surface potential dipole model comparing its performance to the Levenburg-Marquardt combined variable algorithm. Their data was time varying and their measurement set and one set of unknown parameters expanded in proportion to the number of time samples (n_t). They reported that the computation times for the Levenburg-Marquardt algorithm were proportional to n_t^3 whereas the variable projection method was proportional to n_t.

The variable projection method has been used in this thesis and is described in more detail in this section. This application is slightly different as there are up to 12 linear variables in a matrix of uncertain rank and up to 6 non-linear variables. The four possible sets of data (the electric potential and up to 3 magnetic planes) are treated, to an extent, independently, although inter-iteration combination is carried out (see Section 6). The method also was modified according to a subsequent paper by [Kaufman:1975]. A brief comparison is made with the Levenburg-Marquardt algorithm using a quadrupole model which shows the superior speed of the variable projection method.

The method was initially tested using an Excel spreadsheet and then translated into C coding. All the coding for the different models including the coding developed for the Levenburg-Marquardt algorithm is given in Appendix 8. Both methods contain convergence control. The Levenburg-Marquardt algorithm automatically adjusts the control depending on the rate of convergence. The variable projection method only applies the control to the non-linear part of the minimisation. It was better to pre-set a profile for the control due to the sensitivity of the model when processing noisy data and sources close to the surface. Although this implies the possibility of a greater number of iterations, the speed of the method no longer makes this an issue. Any data sets still significantly converging after a set number of iterations (usually between 100 and 200) were rejected.

Symbols used in this section will be similar to those of [Golub and Pereyra:1973] so that any reader making use of this reference will find comparisons easier.

5.1 The Levenburg-Marquardt Algorithm

The least squares metric (or merit function) χ^2 is defined as:

$$\sum_{i=1}^{n} \left[v_i - \sum_{j=1}^{m} a_j \Phi_j(\alpha, t_i) \right]^2 \qquad 5.2$$

α is usually $f(x,y,z)$ the point source cartesian co-ordinates but in the case of two distinct sources could be $f(x_1,y_1,z_1, x_2,y_2,z_2)$. There are m linear variables, each one with its own $\Phi(\alpha, t_i)$. In the dipole case m=3 and in the quadrupole case m=9 although care must be exercised due to the linear dependencies in the equations. However, with noisy data and two or more magnetic planes, it is possible to fit data to all 9 tensor elements. In the following, for simplicity, **a** refers to either any of the m linear variables or to any of the p (3 or 6) non-linear variables α..

The first 3 terms of the quadratic expansion of the χ^2 metric are:

$$\chi^2(\mathbf{a}) = \gamma - \mathbf{d.a} + \frac{1}{2}\mathbf{a.D.a} \qquad 5.3$$

where :

γ is $f(\mathbf{a})$

$$\mathbf{d} = -\nabla f(\mathbf{a})\big|_{\mathbf{a}} = \frac{\partial \chi^2}{\partial a_k} \quad (\text{where } k \text{ is } 1..[m+p])$$

$$\mathbf{D} \text{ (the Hessian)} = \frac{\partial^2 \chi^2}{\partial a_k \partial a_l} \quad (\text{where } l \text{ is } 1..[m+p])$$

D is an (m+p) x (m+p) matrix.:

$$\mathbf{D} = 2\left\{\frac{\partial V(\mathbf{a})}{\partial a_k}\frac{\partial V(\mathbf{a})}{\partial a_l} - [V - V(\mathbf{a})]\frac{\partial^2 \chi^2}{\partial a_k \partial a_l}\right\} \qquad 5.4$$

As will be seen **D** only sets the direction for minimisation and does not influence the final outcome, so it normally truncated to the more simple form:

$$\mathbf{D} \approx 2\left\{\frac{\partial V(\mathbf{a})}{\partial a_k}\frac{\partial V(\mathbf{a})}{\partial a_l}\right\} \qquad 5.5$$

The gradient of χ^2 is:

$$\nabla\chi^2(\mathbf{a}) = \nabla\gamma + \nabla\frac{1}{2}(\mathbf{a}.\mathbf{D}.\mathbf{a} - \mathbf{d}).\mathbf{a} \approx \mathbf{D}.\mathbf{a} - \mathbf{d} \qquad 5.6$$

For small finite steps along the gradient towards a minimum:

$$\delta\left(\nabla\chi^2\right) = \mathbf{D}.\delta(\mathbf{a}) \qquad 5.7$$

so that

$$\nabla\chi^2\big|_{\mathbf{a}\,\text{current}} = \mathbf{D}.(\mathbf{a}_{\text{current}} - \mathbf{a}_{\text{minimum}})$$

or 5.8

$$\mathbf{a}_{\text{minimum}} = \mathbf{D}^{-1}\left(-\nabla\chi^2\big|_{\mathbf{a}\,\text{current}}\right) + \mathbf{a}_{\text{current}}$$

If $\mathbf{D}^{-1}$ is made constant, the equation becomes that of the steepest descent method. The Levenburg-Marquardt method considers the dimensions of $\mathbf{D}^{-1}$ and includes a dimensionless multiplier to the diagonal of **D**. The control of this multiplier controls the step size of the next guess of **a**. So:

$$D_{jj} \rightarrow D_{jj}(1+\lambda)$$

and 5.9

$$D_{jk} \rightarrow D_{jk} \quad (j \neq k)$$

When λ is large, the inversion is diagonally dominant and the equation becomes that of the inverse Hessian method.

By varying λ, the size of the step down the steepest descent is controlled. If the step is too great and $\sum_{i=1}^{n} \chi^2$ is greater as a result then the step is repeated with a larger λ. Zero pivoting during the inverse is usually avoided as small pivots give large corrections which almost invariably overshoot. As λ increases, the modified **D** tends to become positive definite and cannot have small pivots.

The coding for this algorithm is shown in Appendix 8 and is a modified version given in [Vetterling et al:1985]. The modifications are, as mentioned, splitting the linear from the non-linear variables and the method of controlling λ.

5.2 The Variable Projection Method

The typical forward equation for a field point i is (5.1)

$$v_i = \sum_{j=1}^{m} a_j \Phi_j(\alpha, t_i)$$

Note that there are *m* functions of mixed sets (t_i and α) which are wherever possible kept separate so that the *i* sets of t dependent functions are evaluated once only. In the closed form part of the forward solution, separation is not possible and not necessary since the evaluation is very rapid. In the infinite series part of the solution (the magnetic field correction due to the boundary), each element of the computation relating to one element of the Associated Legendre matrix is held in data storage.

The least squares inverse model minimizes the metric functional:

$$r(\alpha, \mathbf{a}) = \sum_{i=1}^{n} \left\| v_i - \sum_{j=1}^{m} a_j \Phi_j(\alpha, t_i) \right\|^2 \qquad 5.10$$

The *n* x *m* matrix $a_j \Phi_j(\alpha, t_i)$ contains $\mathbf{a} \in \Re^m$ and $\alpha \in \Omega \subset \Re^p$

where Ω is an open set containing the desired solution. Note that although *p* contains sets in the same physical space, they are separate and effectively the solution space is *p* dimensional. Searching, then, for two separate sources *p* becomes 6. In the following summation signs are omitted for clarity.

For each α there exists an orthogonal projector, $P_{\Phi(\alpha)}$,on the linear space spanned by the columns of Φ(α) such that:

$$P_{\Phi(\alpha)} = \Phi(\alpha)\Phi^{+}(\alpha) \qquad 5.11$$

The + refers to the pseudo inverse of $\Phi(\alpha)$.

The orthogonal complement is $P^{\perp}_{\Phi(\alpha)} = I - P_{\Phi(\alpha)}$.

For any α (say an initial guess) there is a local minimizer $\hat{\mathbf{a}}$ which satisfies:

$$\Phi(\alpha)\hat{\mathbf{a}} \cong \mathbf{v} \text{ or}$$

$$\hat{\mathbf{a}}(\alpha) \cong \Phi^{+}(\alpha)\mathbf{v} \qquad 5.12$$

Using this (local) minimal solution, the metric functional is modified so that:

$$r(\alpha,\hat{\mathbf{a}}) = \left\| \mathbf{v} - \Phi(\alpha)\Phi^{+}(\alpha)\mathbf{v} \right\|^2 = \left\| P^{\perp}_{\Phi(\alpha)}\mathbf{v} \right\|^2 \qquad 5.13$$

or

$$r_2(\alpha) = \left\| P^{\perp}_{\Phi(\alpha)}\mathbf{v} \right\|^2$$

This modified functional $r_2(\alpha)$ is the variable projection functional and effectively separates the variables. An α is sought, $\hat{\alpha}$,which minimizes this and is then substituted into 5.12 to find a new $\hat{\mathbf{a}}$.

$\Phi(\alpha)$is an n x m matrix where n (the number of data points) $\geq m$ (the number of linear unknowns). The rank, rk, of $\Phi(\alpha) \leq m$ is assumed constant in the open set Ω.

The theorem on which this method is based was put forward by [Golub and Pereyra:1973] and is that, if $\hat{\mathbf{a}}$ is a global minimizer of $r_2(\alpha)$ then $(\hat{\mathbf{a}},\hat{\alpha})$ is a global minimizer of $r(\alpha,\mathbf{a})$ and $r(\hat{\mathbf{a}},\hat{\alpha}) = r_2(\hat{\alpha})$. Also the converse, that if $(\hat{\mathbf{a}},\hat{\alpha})$ is a global minimizer of $r(\alpha,\mathbf{a})$ for $\alpha \in \Omega$, then $\hat{\alpha}$ is a global minimizer of $r_2(\alpha)$ in Ω and $r_2(\hat{\alpha}) = r(\hat{\mathbf{a}},\hat{\alpha})$. If there is a unique $\hat{\mathbf{a}}$ among the minimising pairs of $r(\alpha,\mathbf{a})$, then $\hat{\mathbf{a}}$ satisfies 5.12.

For any given α , Φ can be orthogonally transformed into trapezoidal form. If Φ is of rank r≤rk there exists an orthogonal matrix Q such that:

$$Q.\Phi.S = \left[\begin{array}{c|c} T_{11} & T_{12} \\ \hline 0 & 0 \end{array}\right] \equiv T_0 \qquad 5.14$$

S is a permutation matrix which repositions the most linearly dependent (rk-r) columns of Φ into T_{12} . T_{11} is an r x r upper triangular non-singular matrix. It can always be inverted and inversion is simple and rapid.

An inverse of Φ is:

$$\Phi^{-1} = S\begin{bmatrix} T_{11}^{-1} & 0 \\ 0 & 0 \end{bmatrix} Q \qquad 5.15$$

Q is isometric and this property can be used to restate the linear least squares problem:

$$Q.\Phi.\mathbf{a} = Q.v \text{ or}$$
$$\hat{\mathbf{a}} = (Q.\Phi)^{-}.(Q.v) = S.T_{11}^{-1}(Q.v) \qquad 5.16$$

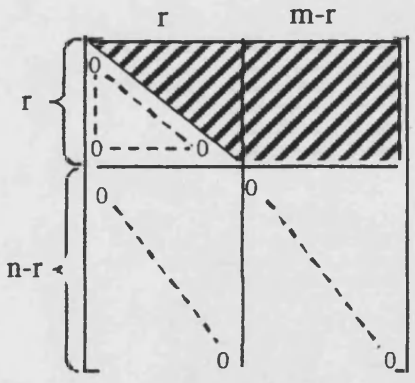
Note at this stage that:

$$\Phi = Q^T S^{-1} T_0 \text{ so}$$
$$P_\Phi = \Phi.\Phi^{-1} = Q^T \left[\begin{array}{c|c} I_r & 0 \\ \hline 0 & 0 \end{array}\right] Q \text{ and}$$
$$P_\Phi^{\perp} = I - P_\Phi = Q^T \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & I_{m-r} \end{array}\right] Q \qquad 5.17$$

Applying Q to v gives an n long vector and applying the first r elements of this (since T_{11}^{-1} only has r non-zero columns) yields the minimal $\hat{\mathbf{a}}$ for the chosen α:

$$\hat{\mathbf{a}} = S.T_{11}^{-1} \left[\begin{array}{c} v_1 \\ \hline v_2 \end{array}\right] \begin{matrix} \}r \\ \}n-r \end{matrix} \qquad 5.18$$

The decomposition of Φ to obtain T_0 and hence $Q.v$ is very simply and speedily done by r Householder transformations. After each transformation, the rms of each column vector is computed and the least significant column is relegated to the right hand end of the matrix. In reality it is not physically placed there but the permutation matrix S is updated. After r transformations the matrix will be trapezoidal:



This method is described in detail in the literature eg [Noble:1976]. The advantage of this transformation is that no large matrix inverses are required and also no large matrix multiplications are necessary as the computation of $Q.v$ is carried out in parallel with the

decomposition. Furthermore, in the next, non-linear, part of the solution the decomposition matrix is required again but only the parameters of the Householder process need be stored, not the whole matrix $\boldsymbol{Q}$.

The non-linear part of this least squares solution uses a variant of the Gauss-Newton-Marquardt algorithm and a simplification pointed out by [Kaufman:1975].

The next $(j+1)^{th}$ guess, α:, is found from;

$$\alpha^{(j+1)} = \alpha^{(j)} + \boldsymbol{K}^{+}_{(j)} \cdot \boldsymbol{f}_{(j)}\left(\alpha^{(j)}\right) \qquad 5.19$$

If there are p non-linear variables then:

$$\text{where } \boldsymbol{f}_{(j)}\left(\alpha^{(j)}\right) = \begin{bmatrix} \boldsymbol{f}_{(j)}\left(\alpha^{(j)}\right) \\ | \\ \boldsymbol{P}^{\perp}_{\Phi(\alpha)} \boldsymbol{y} \end{bmatrix} \qquad 5.20$$

$\boldsymbol{K}^{+}_{(j)}$ is the Moore-Penrose pseudo inverse of a matrix:

$\boldsymbol{K}^{+}_{(j)}$ is the Moore-Penrose pseudo inverse of a matrix:

$$\boldsymbol{K}_{(j)} = \begin{bmatrix} \boldsymbol{D}\left(\boldsymbol{f}\left(\alpha^{(j)}\right)\right) \\ --- \\ v_j \boldsymbol{X}_j \end{bmatrix} \Big\} p \qquad 5.21$$

where $\boldsymbol{D}\left(\boldsymbol{f}\left(\alpha^{(j)}\right)\right)$ is the Frechet derivative of $\boldsymbol{f}\left(\alpha^{(j)}\right)$. In the Golub and Pereyra [*op cit*] paper this evaluates to:

$$\boldsymbol{D}\left(\boldsymbol{f}\left(\alpha^{(j)}\right)\right) = -\boldsymbol{P}^{\perp}_{\Phi(\alpha)} \boldsymbol{D}(\Phi(\alpha))\Phi^{-}\boldsymbol{y} - \left(\boldsymbol{P}^{\perp}_{\Phi(\alpha)} \boldsymbol{D}(\Phi(\alpha))\Phi^{-}\right)^{T} \boldsymbol{y} \qquad 5.22$$

As this defines the size of the step to be taken, the Kaufman [*op cit*] simplification drops the second part of this equation so the evaluation of the Frechet derivative becomes:

$$\boldsymbol{D}\left(\boldsymbol{f}\left(\alpha^{(j)}\right)\right) = -\boldsymbol{P}^{\perp}_{\Phi(\alpha)} \boldsymbol{D}(\Phi(\alpha))\Phi^{-}\boldsymbol{y} \qquad 5.23$$

Now $\Phi^{-}\boldsymbol{y}$ has already been evaluated and is merely $\hat{\mathbf{a}}$ the local minimum vector of the linear variables given $\alpha^{(j)}$: Also $\boldsymbol{P}^{\perp}_{\Phi}$ has already been determined during this evaluation and is available via the stored Householder parameters. $\boldsymbol{D}(\Phi(\alpha))$ is the tensor derivative of each Φ with respect to every α. For example, using the notation of Chapter 3:

$$\boldsymbol{D}(\Phi(x)) = Q_{xx}F_{xxx} + Q_{yx}F_{yxx} + Q_{zx}F_{zxx} + \ldots\ldots\ldots\ldots + Q_{zz}F_{zzx} \qquad 5.24$$

Effectively, the Frechet derivative consists of the octapole terms multiplied by the linear quadrupole coefficients and the derivative is at a minimum when α is at a minimum. This is consistent with earlier views on the location of a dipole [Geselowitz:1960] and some of the quadrupole coefficients [Brody:1968].

Applying $P_\Phi^\perp$ to $D(\Phi(\alpha))$ produces an (n-r) x p matrix. The remaining (r-p) rows are filled with zeros.

The lower part of $K_{(j)}$ consists of a p x p identity matrix multiplied by a control factor v_j which determines the size of the non-linear next step. v_j is chosen so that:

$$\left\| f\left(\alpha^{(j+1)}\right) \right\|_2^2 \le \left\| f\left(\alpha^{(j)}\right) \right\|_2^2 \qquad 5.25$$

although with a fixed profile for v_j as has been mentioned, this is not always the case due to the highly non-linear nature of the equations.

Once the matrix $K_{(j)}$ has been formed, its pseudo inverse is found in the same way as $P_\Phi^\perp$ but this time by p Householder transformations. The permutation matrix S is not required as the non-linear case is full rank.

5.3 Computational Comparison

The two methods were compared using the surface potential generated by an arbitrarily located point quadrupole source. Data was generated by a single run of the forward model for 87 equi-spaced points on the surface of a sphere with a radius of unity. There are 5 independent quadrupole coefficients and these were arbitrarily chosen as:

$$Q_{xx} = 0.05 \quad Q_{yx} = 0.054 \quad Q_{zx} = 0.06$$
$$Q_{yy} = 0.057 \quad Q_{zy} = 0.04$$

Two comparison runs were made, one with the source located at (0.1,0.1,0.1) 17% of the radius from the centre, and the other at (0.5,0.5,0.5) 87% of the radius. Both methods found the second, shallower, source more difficult to locate. The starting assumptions for source location were well removed from the true source being (0.5,0.35,0.5) for the deep source and (0.1,0.1,0.1) for the shallow source. The quadrupole strengths were always started at unity:

	Variable Projection	Levenburg-Marquardt
Deep Source:		
End point reached		
Location	(0.1,0.1,0.1)	(0.1001,0.0994,0.0991)
Quadrupole strengths	(0.05,0.04,0.06,0.07,0.04)	(0.05,0.04,0.06,0.07,0.04)
Computation time	app 5 seconds	app 10 seconds
	6 iterations	19 iterations
Shallow Source:		
End point reached		
Location	(0.5,0.5,0.5)	(0.52,0.45,0.39)
Quadrupole strengths	(0.05,0.04,0.06,0.07,0.04)	(0.02,0.06,0.17,0.11,0.17)
Computation time	app 8 seconds	app 5 minutes
	10 iterations	>250 iterations

In the Levenburg-Marquardt run with the shallow source, the run was terminated before full convergence and might well have converged if allowed to continue.

5.4 Conclusion

There is little doubt that the variable projection method is superior to the slower Levenburg-Marquardt algorithm. This conclusion is in line with the experience of [Oostendorp and van Oosterom:1993]. An advantage of the variable projection method is that rank deficient matrices can be handled more easily especially if there is some uncertainty as to the degree of that deficiency as the selection of the most linearly dependent column vectors can be made automatically and indeed can vary from iteration to iteration.

Both methods still require the differentiation of the non-linear functionals and both methods are able to control the rate of convergence, which is particularly important in non-linear minimizations where the topography is uncertain and irregular. The speed of the variable projection method is mostly due to the avoidance of large matrix inversions.

The search for a fast least squares algorithm was originally motivated by the need to handle efficiently large and involved equations. However, this part of the least squares process is much the same whichever method is chosen and speed is gained here from efficient coding. It became apparent that the number of iterations required would be a significant factor when the equations were used to simulate a real world situation (eg shallow sources and the presence of noise) so the variable method was adopted for all models.

6. Modelling and Results

The objective of modelling the analytical solutions in Chapter 3 is to explore the behaviour of the equations when they are used in a manner which approximates the real world. As pointed out in Chapter 1, previous studies have indicated the difficulties of accurate source localization and of attributing inaccuracies to specific causes out of many possibilities. The analytical solution allows some of the causes to be studied in isolation but it must be noted that:

1. an analytical solution is itself an approximation since many assumptions are made in the course of its derivation.
2. there may be many other factors in the real world that are sources of error equal to or greater in magnitude than the analytical sources or even the possibility that some may cancel out others.

The questions which this Thesis seeks to answer and, where possible, quantify are:

- is it possible to detect the location of a quadrupolar current source of unknown strength through inverse modelling of the surface fields it generates:
  - a) in the presence of a stronger nearby or coincident dipole current source
  - b) as for a) but with the addition of a realistic amount of signal noise

- what is the magnitude and direction of localization error:
  - a) of a dipole in the presence of a nearby unknown quadrupolar current source
  - b) of a quadrupolar current source in the presence of noise

- what is the sensitivity of the model to sources at different depths? Is the resolution reduced (ie. increased magnitude and direction error) as sources become deeper?

- for all the above, what is the difference between inverse modelling the electric potential and the magnetic field? Is there any advantage in combining the two types of field measurement and what is the best way to combine them?

There are other questions which could be addressed from analytical modelling. These are not answered in this Thesis as they are felt to be of lesser importance than the principal questions above. This work could be fairly easily extended to cover them:

- what degree of field measurement sensor location could be tolerated in detecting quadrupolar sources?
- what is the effect of omitting the correction for the volume current terms in the magnetic inverse model?
- the models are extendible to cover additional point sources, so what is its ability to separate three or more quadrupolar and/or dipolar sources
- by modelling the prolate spheroid equations, what is the effect of curvature errors from assuming a spherical solution?

6.1 Computer Model Development

A suite of computer models was developed stage by stage with verification at the end of each stage:

- surface electric potential
  - the forward dipole model
  - the forward quadrupole model
  - inverse models for each of the above
  - combined dipole and quadrupole inverse models (two versions: coincident and non-coincident sources)
- magnetic field modelling
  - the infinite medium forward models
  - the boundary correction forward models
  - the separate inverse models
  - combined dipole and quadrupole inverse models (also two versions: coincident and non-coincident sources)
- Electric and Magnetic Field Models
  - dipole, quadrupole and combined inverse models
- Other
  - verification models
  - data generation models

6.1.1 Hardware and Software

All models were developed in standard ANSI C using a Symantec C++ compiler. Some minor C functions were taken from [Vetterling et al: 1985], however these had to be modified to fit into the model. All other functions were original.

The mathematics package Maple V was used to check for differentiation errors during equation development, and for plotting and rendering results. Excel spreadsheets were used for data generation and results' summaries and graphing. Excel spreadsheets were also powerful enough to assist with model verification as all inverse models (except for the magnetic field boundary correction models) were initially developed using them. As the larger C models were developed, intermediate values generated by the model could be cross checked. The nature of the inverse solution meant that the model was totally intolerant to even the slightest error.

A freeware package called Dome was downloaded from the Internet in order to generate equi-spaced co-ordinates on the surface of a sphere. This package is written in C++ and uses geodesics to generate co-ordinates, wireframes and other output. It was modified to fit in with the requirements of this model but in the event was only used in the development stage to generate sets of co-ordinates covering an entire sphere.

All development and model results were done on a Macintosh G3 266MHz Powerbook. At the time of model development this was one of the faster laptop computers and proved adequate for the task. Run times for statistical testing took up to 20 hours and consisted typically of 50 sets of 200 iterations of three parallel inverse models (56 field measurement points per model).

6.1.2 Principal Control Variables

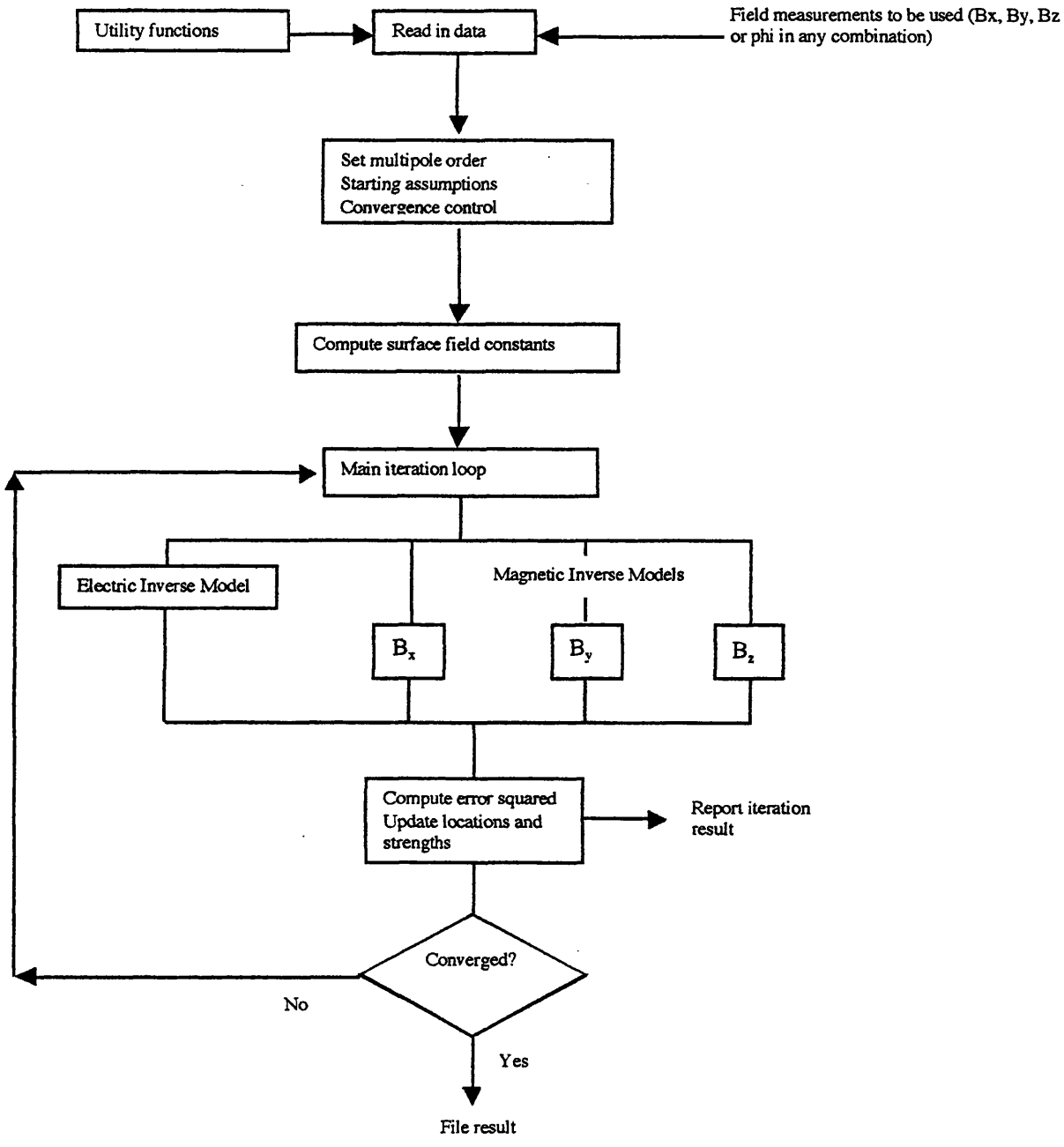
Surface co-ordinates	may be the same or different in both number and position on the basis that this is likely to be the case for real world measurements (eg more electrodes than gradiometers and different positions)
Multipole order	infinite multipole series can be truncated at any point in the model. It was set to 7 for all modelling work.

Field measurements	generated by a single forward model runs. Random noise added at different SNRs. Co-ordinates and data read in from file. The number of measurement planes used in the inverse model is selected at run time.
Convergence control	the upper Cholesky factor is set as a profile which decreases after a variable number of iterations. The profile is set up by trial and error based on the highest convergence speed consistent with stability and the fewest number of convergence failures.
Rank	Strictly speaking rank is not a variable but it can be reduced from a single variable if rank deficiency becomes a problem.
Starting point	location of first guess of source which should be somewhere in the equivalent cardiac region. Convergence point should be independent of starting point but if the topography is difficult this may not always be the case as the model settles on a local minimum. Dipole and quadrupole component strengths are always started at unity.
Stopping point	1) A fixed number of iterations and/or 2) A minimum least squares error is reached and/or 3) A minimum least squares error change between successive iterations

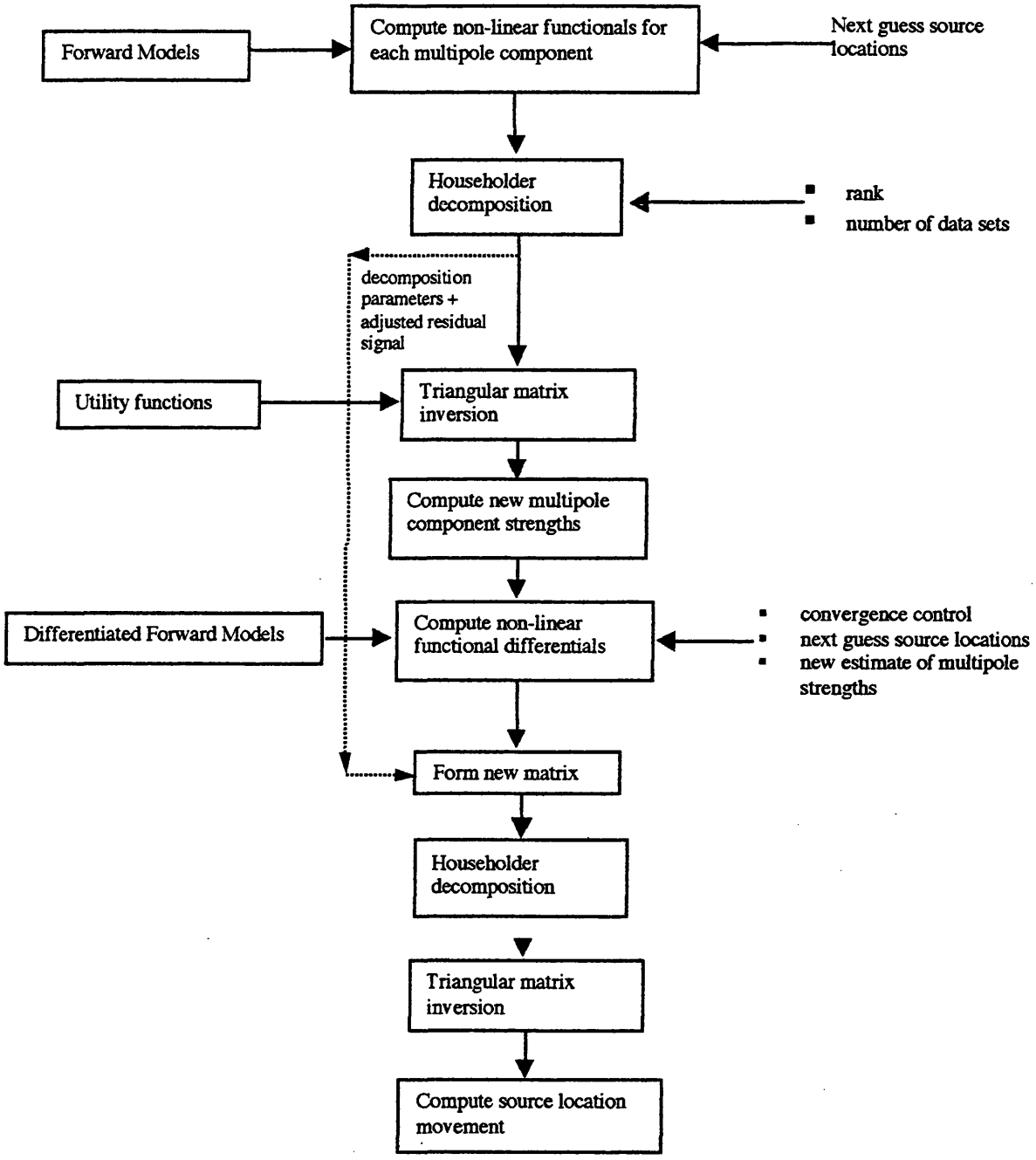
6.1.3 Model Layout

Details of each model are shown in Appendix A8. The following is a generalised block diagram for an inverse model.

Outer Control Loop



Variable Projection Inverse Model



6.1.4 Model Verification

Although a large number of published papers have used the spherical model, direct comparison of the output of the forward model in this Thesis has not been possible:

- field data is often not reported or reported only partially.
- often comparative surface maps are shown but usually with normalised or relative scaling or none at all.
- key input data is sometimes missing (eg dipole strengths or depths).
- most models are dipole models and based on a limited expansion of the space vector, not the closed form used in this Thesis.

In order to verify that the output from the models is sensible, a more indirect approach was adopted. Firstly, can the magnitude of the output be related to output reported in other papers?

There are two constants in the equations: σ an average conductivity for the torso, μ_0 free space magnetic permeability ($4\pi x10^{-7}$ v/Am). A typical average conductivity is 0.22 S/m.

Dipole and quadrupole strengths and their resultant surface electric potential fields are widely reported in the literature and typical fields are in the range 0 – 1 mv. Magnetic field measurements are more difficult to find and often published papers contain incomplete assumptions. The following table shows some of the reported data:

	Current Dipole	Current Quadrupole or magnetic dipole	Observed or calculated field	Source depth
Karp et al:1980	-	750 nAm^2	-	-
Erné ,Trahms et al:1987	5600 nAm	200 – 700 nAm^2	20 – 50pT	14 cms
Ascenbach, Mosage et al:1992	to 2000nAm	-	-	-
Geselowitz & Miller:1973	10000nAm	-	-	-
Geselowitz: 1980	-	300 –700 nAm^2	-	-
Gonelli & Agnello:	800 –7200 nAm	24 – 1200 nAm^2	-20 – 80 pT	7 cms
Einolo, Nenonen et al:1993	-	-	0 - 13 pT	5 to 13.5 cms
Killmann, Weismuller et al :1995	30 – 260 nAm	-	0.7 – 3.2 pT	15 cms

Assuming a current dipole strength of 5000 nAm, the dipole models gave the following results on the surface of a sphere:

Source Depth	Electric Field		Magnetic Field	
	mean	peak	mean	peak
5 cms	0.14 mv	1.07 mv	20 pT	108 pT
15 cms	0.03 mv	0.06 mv	3.9 pT	5.5 pT

The quadrupole is more difficult to verify due to the paucity of published data. Assuming a current quadrupole strength of 110 nAm^2 (about –33dB strength relative to the dipole) the quadrupole forward models gave the following result:

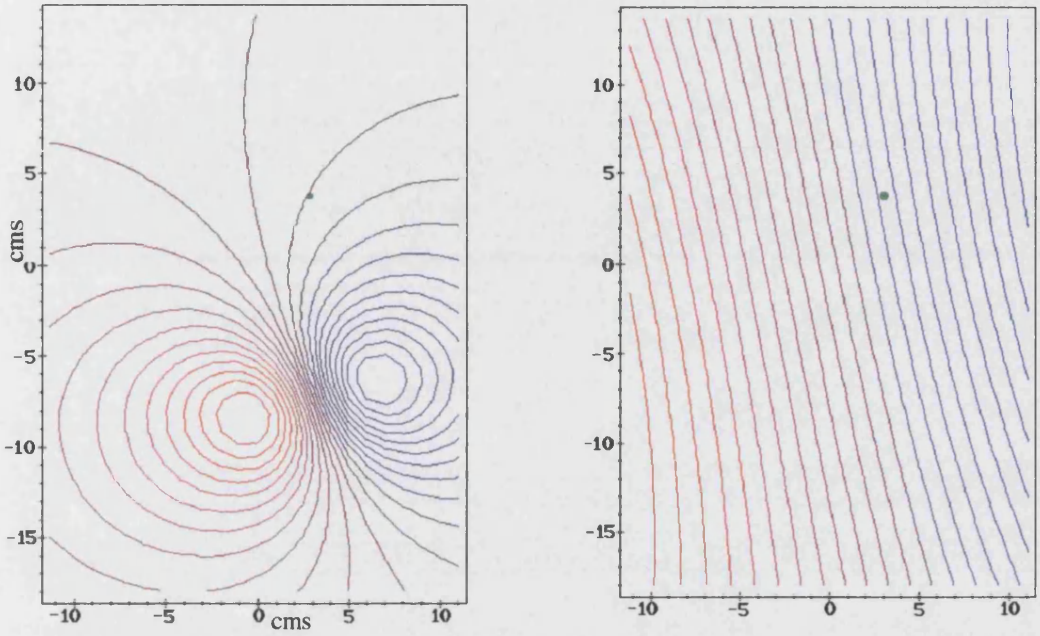
Source Depth	Electric Field		Magnetic Field	
	mean	peak	mean	peak
5 cms	0.07 mv	0.9 mv	10 pT	66 pT
15 cms	0.004 mv	0.013 mv	0.8 pT	1.3 pT

The next stage in verifying the model is to examine contour maps of the surface data generated and compare them with published maps. Figs 6.1 to 6.4 show the magnetic field patterns for a dipole and quadrupole source. Figs 6.1 and 6.2 show the effect on the infinite medium magnetic field of volume currents at the boundary. As can be seen the net magnetic field adjustment is small which confirms the view of [Horacek:1987] and [Purcell:1988] (see below) in that it has little effect on the shape of the infinite medium pattern. The magnetic patterns are similar to those seen in [Karp et al:1980] which are from fitting real patient data using an infinite medium spherical expansion and magnetic dipoles as the source. Also a comparison can be made with patterns in [Erné, Trahms et al:1987] who also use an infinite medium model but a half space rather than a sphere. They examined all the terms of the quadrupole tensor.

The quadrupole is combined with the dipole and Fig 6.3 shows the result. The dipole to quadrupole signal strength was 33dB and the effect of the quadrupole is to distort slightly the dipole pattern. Finally the effect of white noise at 15dB SNR is shown in Fig 6.4. The signal is somewhat broken up and although the dipole pattern can be discerned, the quadrupole pattern is not visible.

Surface Magnetic Field from a dipole source

Infinite medium field less Conductivity boundary field



gives: Combined Field

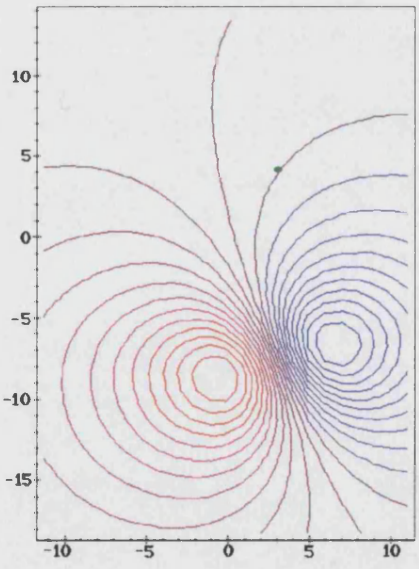


Fig:6.1 Magnetic field (B_y) pattern on the surface of a small part of a sphere from a point dipole • 5.2 cm below the surface.

Surface Magnetic Field from a quadrupole source

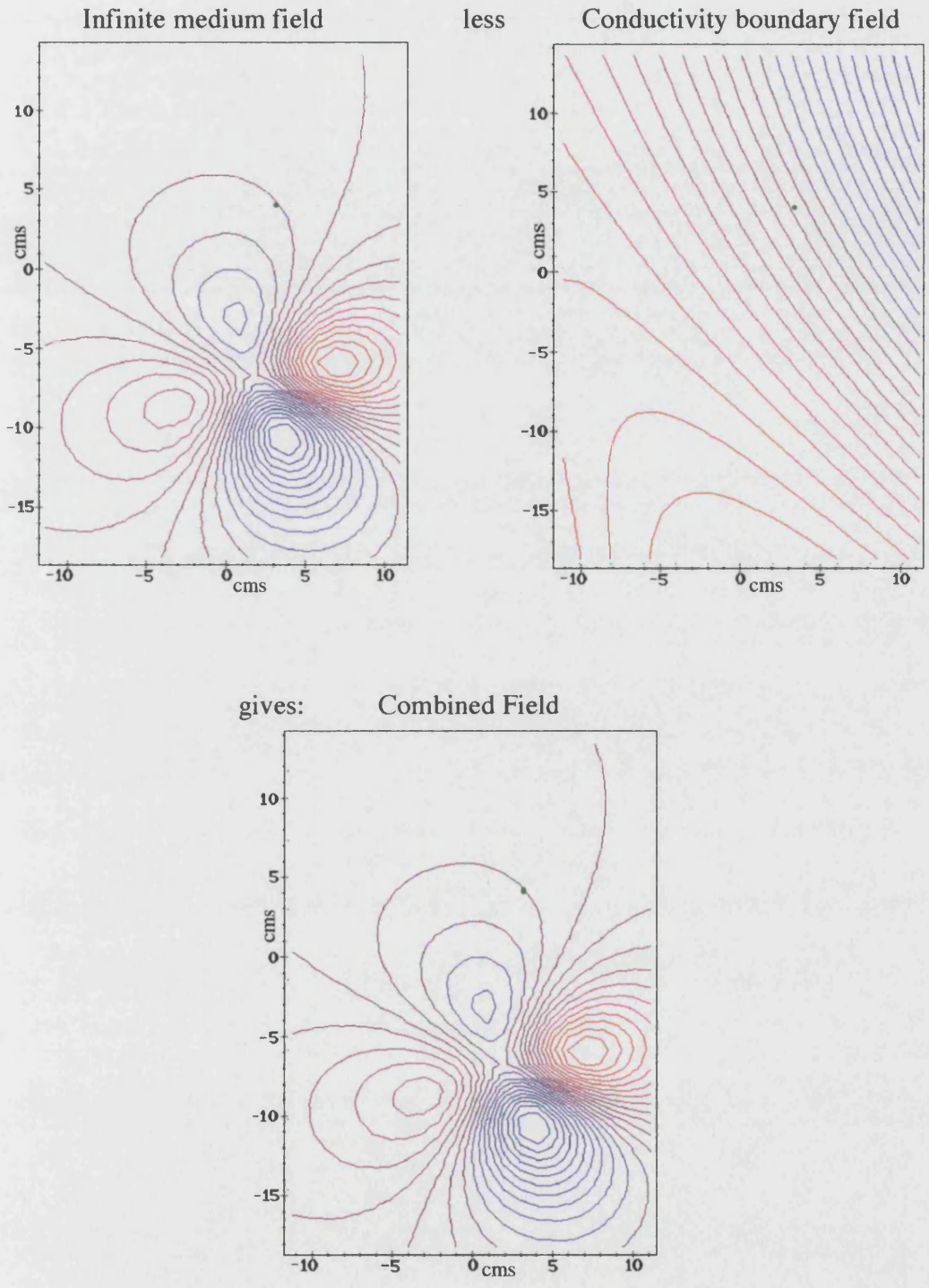
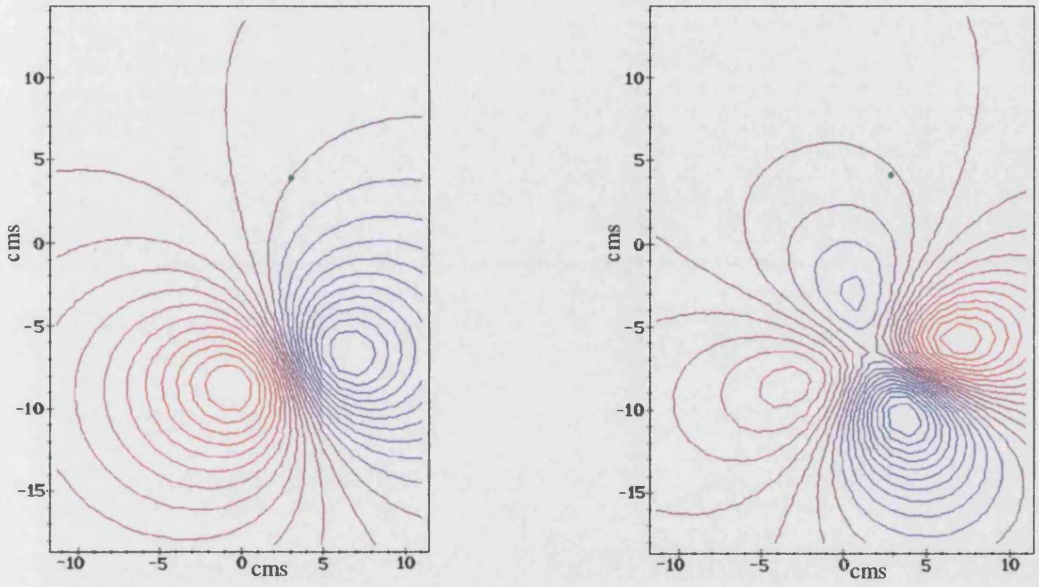


Fig:6.2 Magnetic field (B_y) pattern on the surface of a small part of a sphere from a point quadrupole • 5.7 cm below the surface.

Dipole plus Quadrupole Surface Magnetic Fileld

Dipole plus Quadrupole



gives:

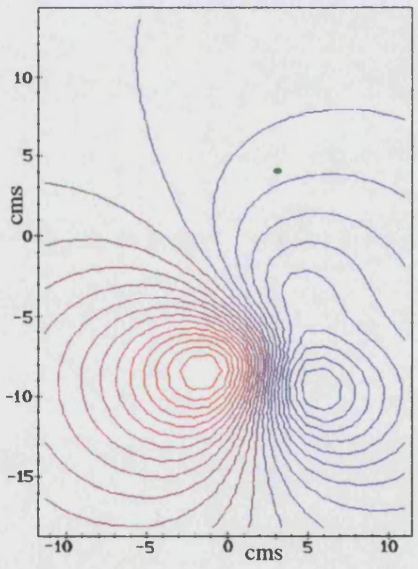
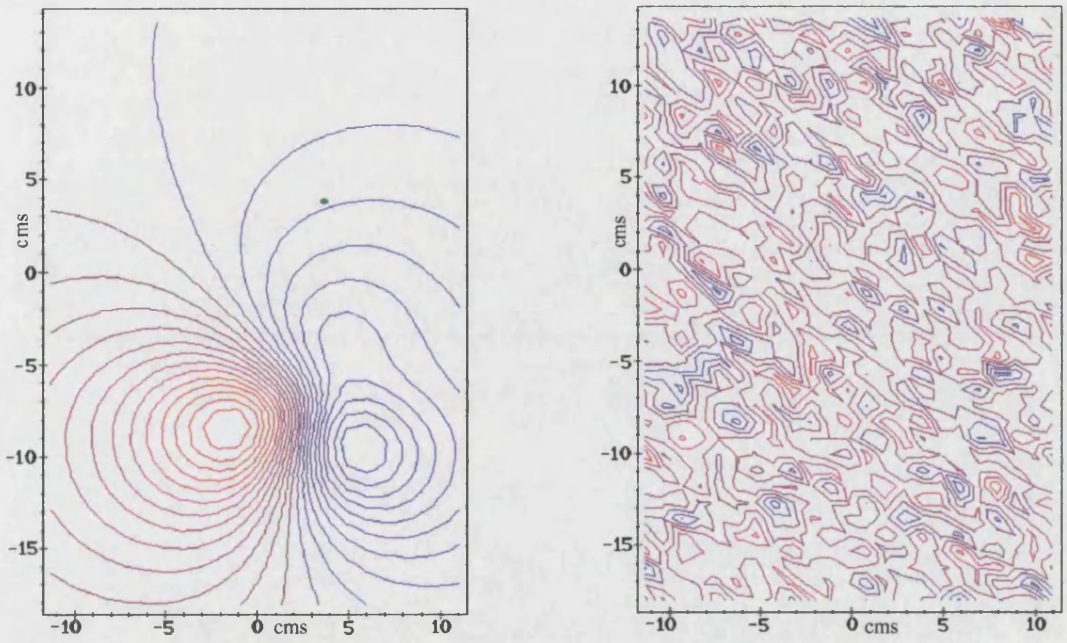


Fig:6.3 Magnetic field (B_y) pattern on the surface of a small part of a sphere from a point dipole and quadrupole • 5.2cm and 5.7 cm below the surface 33dB relative strength.

Dipole + Quadrupole + Random Noise at 15db SNR



gives:

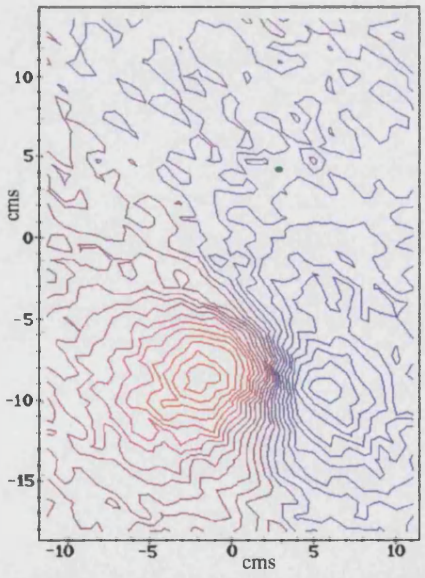


Fig: 6.4 Magnetic field (B_y) pattern with 15dB SNR random noise added. Point Dipole and Quadrupole source at 33dB relative strength.

Not many authors have produced maps or data examining the effect of the volume currents. However, [Nenonen:1991] reported studies by [Horacek:1987] and [Purcell:1988] who concluded that the dominant features of dipolar activity were preserved in the maps. They also reported that the body surface had a more significant impact than internal inhomogeneities. The volume currents serve to weaken slightly the infinite medium or direct magnetic field. Also [Zimmermann & Erné:1991] conducted a study on a sphere and the effect of volume currents. They were investigating the impact of sphere deformation on the magnetic field and reported that, in general, volume currents resulted in an overall field magnitude reduction of 10% with larger localised variations but a dipolar type of field pattern. The field extrema were also further apart than the extrema of the 'direct' dipole. The volume current influence increased with source depth (due to the reduction of the 'direct' influence). The model developed here shows similar features, however, Fig 6.5a shows the B_y total field pattern superimposed on the infinite medium field for the shallow dipole case and the deep case. The peaks have been displaced slightly but remain the same distance from each other. This is possibly due to the symmetry of the spherical model which causes a very even field from the volume currents. The right hand diagram in Fig 6.5a also shows the field pattern from a deeper dipole which is much the same but, as could be expected, with peaks further apart. Note that although the polarity seems to be same as the 'direct' source for the dipole case in Fig 6.1, in the model the volume field is subtracted from the 'direct' field so that it is effectively opposite polarity and serves to weaken the infinite medium field. The average reduction in field magnitude is 10% but the extremes of the range are –63% to +69%.

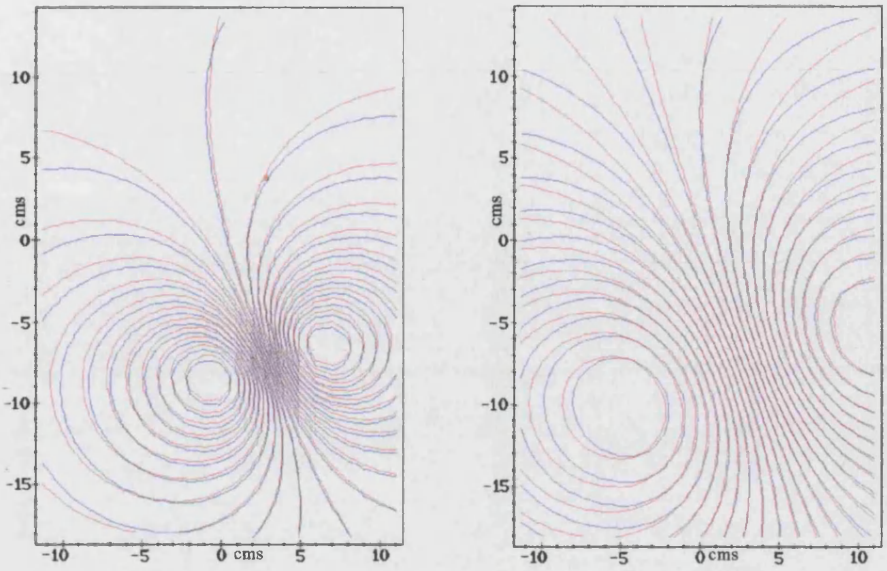


Fig 6.5a Magnetic field (B_y) for a shallow dipole (5.7cm – left diag.) and a deep dipole (11.1cms – right diag). Infinite medium (red) superimposed on total field(blue).

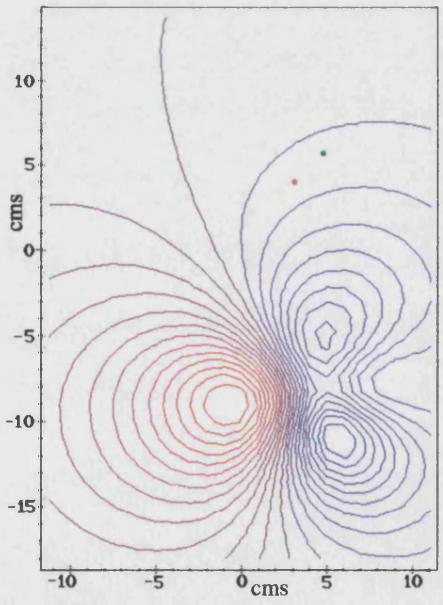


Fig 6.5b Total Magnetic Field (B_y) from a dipole • and a quadrupole • 5.1cms (avg) depth, 2cms apart.

The effect of laterally separating the dipole from the quadrupole produces a slightly different picture Fig 6.5b with three peaks. The field can be drawn in 3-dimensions but serves no particular purpose other than illustration. Fig 6.6 shows the infinite medium vector field from a shallow dipole drawn for four different quadrants of the measurement grid for easier viewing. As can be seen, the B_y component matches the contour diagram in Fig 6.1.

A final check in the verification process can be made by virtue of the integration used to derive the volume current magnetic field. The integration around a closed sphere of the magnetic field in any one orthogonal plane should sum to zero. Equi-spaced points were generated across the entire sphere surface and both the dipole and quadrupole point source volume models were run with multipole order 7. All three planes individually summed to a figure close to zero. This proved to be a valuable guide when debugging the software.

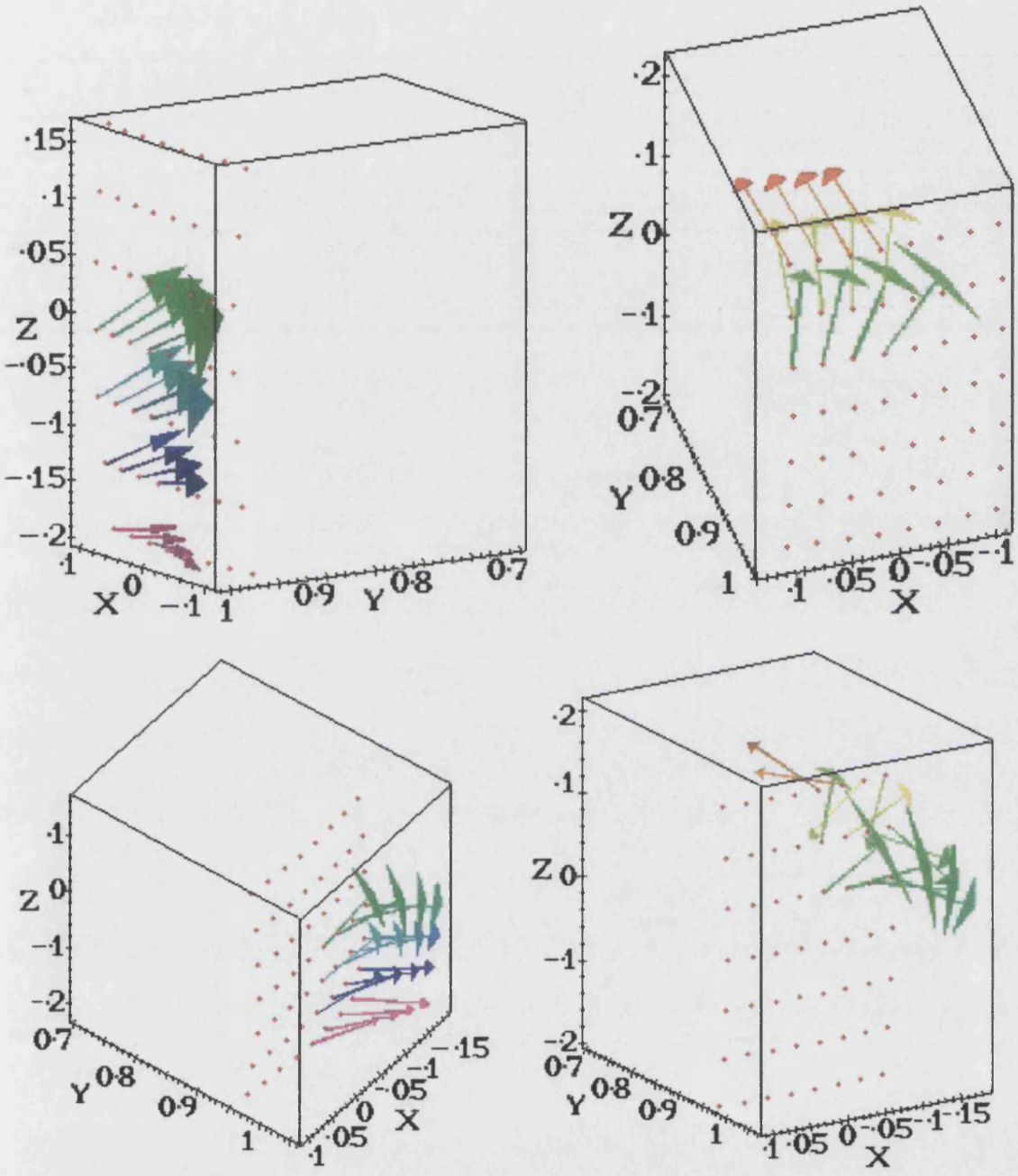


Fig 6.6 Vector picture of the surface infinite medium magnetic field from a shallow [0.03,0.95,0.04] dipole. Field is split into quadrants for ease of viewing. Arrow head width is proportional to field strength

6.1.5 Inverse Model Characteristics

As a result of developing and running the various inverse models, their characteristics and sensitivities become apparent. Some of the main features are described here.

6.1.5a Rank Deficiency

The method of separating the linear from the non-linear variables described in Chapter 5 leads to an understanding of the linear and near linear dependencies in the model.

There are a total of 12 possible linear variables (3 dipole and 9 quadrupole components) and, in the case of separately located point dipole and quadrupole sources, 6 non-linear variables (the xyz co-ordinates of each source). Note that the quadrupole components, according to theory, are not all independent and the model can, optionally, enforce this dependence. If the original data was generated with two of the sources in the same position, the non-linear part of the model, which has rank 6,will eventually fail as the upper triangular matrix will contain singularities. If the rank is reduced to 3, a solution will always be found. As it is not possible in the real world to anticipate whether the two sources are coincident, two models were written. This Thesis examines known data so the appropriate model is always selected. With unknown data, the first model to use would be the separate point source model and only in the event that repeated singularities were encountered due to coincident sources would the rank 3 model be selected. However, if the patient diagnosis indicates otherwise, then the separate source model could be used. For separate point source data, it was noted that if the model is initialised with starting locations swapped over and, during a sequence of iterations, a cross-over happens, a singularity has not yet occurred (after some 5000 runs of between 100 and 200 iterations).

In the case of the linear variables there are two parts of the model to consider – the infinite medium and the volume current calculations:

The infinite medium - the electric potential model has rank 8

D_x, D_y, D_z, Q_{xx}, Q_{yx}, Q_{zx}, Q_{yz}, Q_{yy}

After convergence Q_{yx}, Q_{zx}, Q_{yx} are twice their correct values (since, $Q_{yx}= Q_{xy}$, $Q_{zx}= Q_{xz}$,$Q_{yz}= Q_{zy}$) and Q_{xx} and Q_{yy} can be derived from the model's answers Q'_{xx} and Q'_{yy} since Q_{zz} is distributed between them (the trace of the quadrupolar tensor vanishes: $Q_{xx}+Q_{yy}+Q_{zz}= 0$):

$$Q_{xx} = (2Q'_{xx} - Q'_{yy})/3$$
$$Q_{yy} = (2Q'_{yy} - Q'_{xx})/3$$

a) the magnetic model has rank 7 so that 7 independent linear variables can be derived depending on which plane is being computed:

B_x	Q'_{xx}	-	-	-	D_y	D_z
	Q_{xy}	-	Q_{zy}			
	Q_{xz}	Q_{yz}	-			

B_y	Q'_{xx}	Q_{yx}	Q_{zx}	D_x	-	D_z
	-	-	-			
	Q_{xz}	Q_{yz}	-			

B_z	Q'_{xx}	Q_{yx}	Q_{zx}	D_x	D_y	-
	Q_{xy}	-	Q_{zy}			
	-	-	-			

Q'_{xx} contains differing amounts of Q_{xx} and Q_{yy} and can only be derived if 2 or more magnetic planes are computed:

from the x and y planes:

$$Q_{xx} = (4Q'_{xx}[y] - Q'_{xx}[x])/3$$
$$Q_{yy} = (Q'_{xx}[x] - Q'_{xx}[y])*2/3$$

from the x and z planes:

$$Q_{xx} = Q'_{xx}[x] + 2Q'_{xx}[z]$$
$$Q_{yy} = (Q'_{xx}[x] - Q'_{xx}[z])/3$$

from the y and z planes:

$$Q_{xx} = (Q'_{xx}[y] + Q'_{xx}[z])/3$$
$$Q_{yy} = (Q'_{xx}[y] - Q'_{xx}[z])*2/3$$

and in all cases $Q_{zz} = -(Q_{xx}+Q_{yy})$

The volume currents - These magnetic fields are derived from the magnetic vector potential which contains all the information of the linear and non-linear variables and so has rank 12. It is possible therefore to

derive all 12 linear variables. In practice there are near linear dependencies but these do not seem to cause a problem with the variable projection method although in the cases where the volume current contribution to the total field is small (shallow sources) the convergence rate is very slow. However, rank can be reduced in the model leaving it to select the least linearly dependent columns at each iteration. The result is a faster but less accurate convergence. However, see the results below.

6.1.5b Initialisation

The data settings which are chosen to initialise the various inverse models should not affect the outcome (the minimum least squares error). However, the models are complex and contain many competing variables. No prior knowledge is assumed concerning multipole strengths so these are set to unity even though they might finish with exceedingly small values. One iteration usually resets most of them to completely different values and in the case of near linearly dependent coefficients to relatively large numbers before they are brought down to their final values. Whatever they are initialised at there is no impact on the final outcome.

In Section 6.2 a description of the data generation is given and a notional volume is defined as the cardiac region. The initial guess for the source location is put somewhere in or near this region. However, even if it is put well outside this region, the models usually converge towards the true position.

Source data and co-ordinates are read in from file. Co-ordinates can be different in number and location, or the same, or a mixture. One or more measurement sets (that is phi, B_x, B_y, B_z) is selected for error squared minimization.

The convergence control profile (the Cholesky positive definite matrix factor) is set. The electric and magnetic unification method is chosen and finally, before the inverse is started, the end point criteria are set.

6.1.5c Sensitivities

As a result of running the models, especially with data containing high levels of noise, three end point sensitivities became apparent:

- Initial Position

On occasions, the starting position could influence the end point. This occurred when there were adjacent minima of sufficient 'depth' (ie surrounded by least squares steep slopes) that the model was unable to escape to the global minimum. Increasing the convergence rate would allow a sufficient jump to be made out of the hole but could then cause instability.

- Convergence Control

The rate of convergence is one of the key controls in the behaviour of the model. Each data set (that is phi, B_x, B_y, B_z) has its own control profile. One method adopted in the Levenburg-Marquardt method is to alter the control depending on the rate of convergence or divergence of each iteration step and attempt to keep to a uniform rate. However, the highly non-linear behaviour of the error squared measure made this a less satisfactory way than setting out a fixed profile related to the number of iterations. The profile was initially determined through trial and error, and once set usually covered a wide range of data sets. The rate of convergence should not affect the end point of the minimization but this was not the case for combined dipole and quadrupole models. This sensitivity is discussed in subsequent sections but to illustrate the point a series of model runs was conducted.

The tests are to show the ability of the model to balance competing data sets. If measurement and noise errors in the electric and magnetic data sets indicate to the inverse model two different positions for, say, the same dipole, in an ideal model the combined inverse should come up with a midway point.

A set of surface data was generated by a single run of the magnetic forward model for a dipole at a deep position relative to the surface:

dipole at (0.03,0.828,0.04)

The electric potential forward model was then run for the dipole but with its position displaced:

electric dipole at(0.038,0.836,0.048)

They should of course coincide as they are the same source. The combined electric and magnetic inverse model was then run to its end point to determine where the resultant position was placed, but using different convergence rates in the electric part of the

inverse from the magnetic part. The rates used were 1 (highest rate) and 10. As seen in Fig 6.7 the resultant was always the mid point for different co-ordinates (x,y,z):

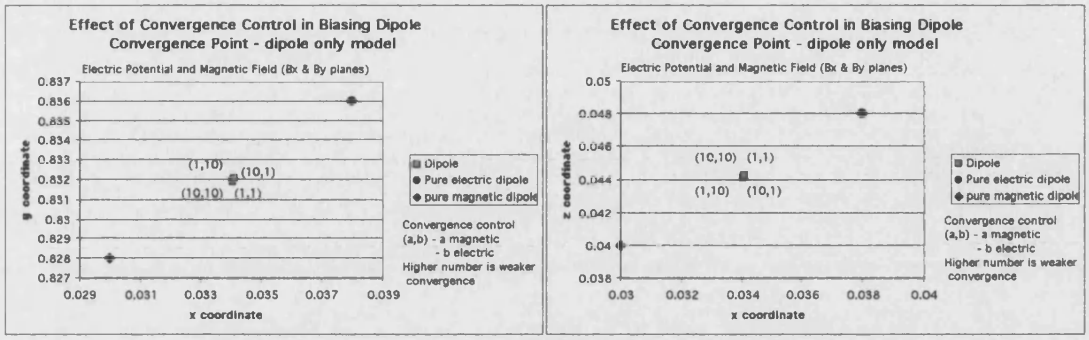


Fig 6.7 Combined electric and magnetic inverse models. Dipole only but electric and magnetic positions displaced.

Similarly the electric potential and magnetic forward models were run for the quadrupole also with positions slightly displaced:

electric at (0.038,0.831.0.048) magnetic at (0.03,0.823,0.04)

The quadrupole (not shown) also gave the mid point for all co-ordinates.

However, the combined dipole and quadrupole inverse model, using the above, data gave different answers, Figs 6.8 and 6.9:

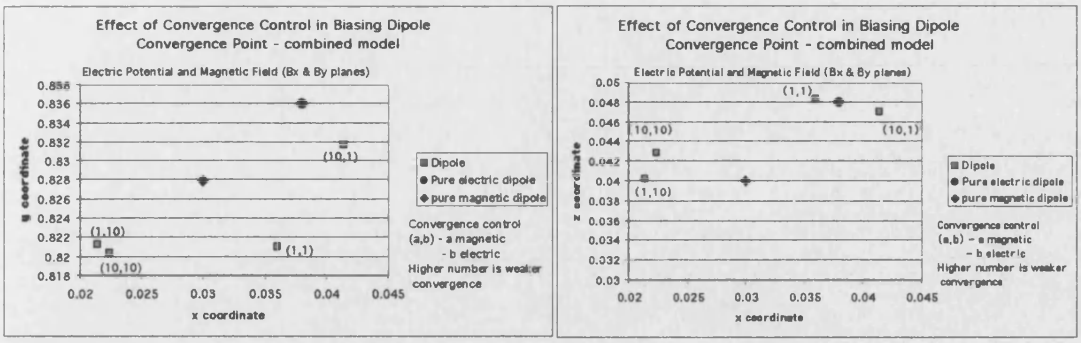


Fig 6.8 Combined electric and magnetic inverse models. Electric dipole and quadrupole displaced from magnetic. Impact on dipole location

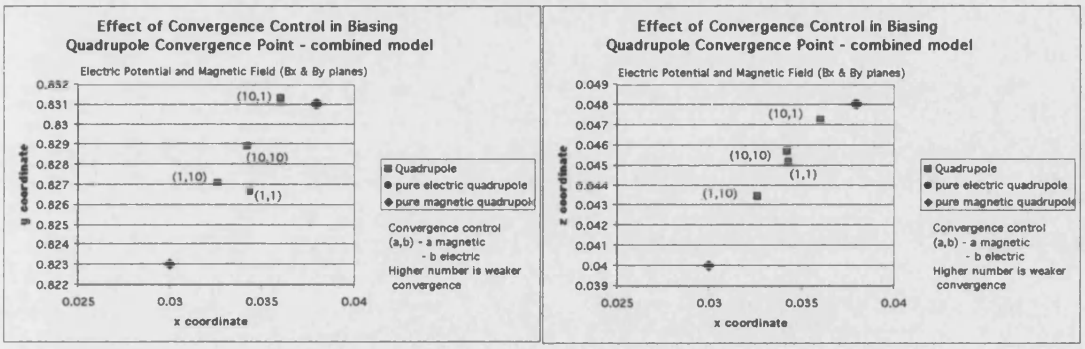


Fig 6.9 Combined electric and magnetic inverse models. Electric dipole and quadrupole displaced from magnetic. Impact on quadrupole location

The above figures 6.8 and 6.9 show the following:

a) There is some interaction between the dipole and quadrupole in that the dipole is displaced to positions which (except for the z co-ordinate) are not between its extremes.

b) The convergence rate affects both outcomes but less so in the quadrupole case. This result foreshadows other results later in this chapter.

c) Providing the rates are the same or the magnetic rate is faster, the resulting quadrupole is close to its mid point position.

- Multiple Minima

  Two types of end point local minima were encountered:

  1) where a minimum was reached from which the model could not escape to a nearby deeper minimum. This happened, for example, when the x and z co-ordinates were the same for each source and the y co-ordinate of one

source ended up on the wrong side of the other source. The model was unable to reverse the positions unless the convergence control allowed a sufficiently large step to be taken and, as mentioned above, this could lead to instability. One method of testing whether or not the minimum reached was a true global minimum was to input a step impulse to the local position co-ordinates after a fixed number of iterations and see whether the model returned to its original minimum or some new and better nearby minimum. There seems to be no way other than trial and error of determining where the true global minimum lies.

2) where two nearby minima existed which were about equal in magnitude, the model could oscillate from one to the other. This happened when two or more measurements sets were used. A small change to the convergence control setting caused the model to settle on a final position.

6.1.6 The Unification of the Electric and Magnetic Fields

It is thought that a method which combines the two data types could yield a better localization result [Nenonen:1992]. This is based on the theory which states that (for the infinite medium case) the origins of the two fields $\left(\nabla \bullet \bar{J} \text{ and } \nabla \times \bar{J}\right)$ are mathematically independent.

Combining the two inverse solutions should give some sort of 'centre of gravity' estimate of the source strength and location. The method proposed by [Hasson:1991] attempts to do a combined minimization using a factor to balance the contributions to the error squared metric from each type of measurement (due to the different measurement scales produced by the same source increments). The method also incorporates a regularisation parameter to smooth inconsistent data.

This Thesis is an investigation into point sources and, if each model has a different view as to the location (and strength) of each source due to uncertain (for example, noisy) data, then some set of weighting criteria have to be found which reconcile the differences. To make a rational choice of weighting factor, an understanding of what causes this difference, and its extent, is needed. An analytical model , although not realistic, offers some insight into the causes and an opportunity to quantify them.

As shown in Section 5, the variable projection model is driven by the differential of the non-linear functionals with respect to each source co-ordinate. The linear variables merely minimise the error for a given location and, although in practice computed first in the procedure, strictly speaking come after the computation of the new location. Any scaling factor applied to the original data is effectively the application of a linear operator and, if everything else is equal, is the same as an alteration to the source strength.

The non-linear functionals for each measurement type are each multiplied by a linear variable. They are equations containing sets of various combinations of the source and field co-ordinates. It must be assumed that in the real world the field measurement points are different in number and position. As far as the least squares minimization is concerned the field co-ordinates, which are kept separate, evaluate to constants.

The question arises as to whether it makes sense to combine the different equations into a single minimization and mathematically how to accomplish this. Although they are mathematically

independent, they map different measurements into the same solution space. The objections to doing a combined (weighted) minimization are:

- The magnitude and sensitivity of the mapping would have to be adjusted so that the correct weight is given to each one. However, without a priori information of which mapping is nearest the correct solution, the weighting would have to be equal.
- The rank of the linear variables is different. This is particularly evident in the case of electrically silent magnetic sources. For example:

$$Q^e_{xy} = Q^e_{yx} \quad \text{and} \quad Q^m_{xy} \neq Q^m_{yx}$$

$$\text{but} \quad \frac{Q^m_{xy} + Q^m_{yx}}{2} = Q^e_{xy}$$

For these reasons and also because this Thesis explores the process of unification, all minimizations are kept parallel and a number of options can be set between each iteration to force certain outcomes:

- The new locations for the dipole from the magnetic and electric potential measurements are averaged – the same for the quadrupole. If two or more magnetic planes are used, the new locations are averaged first before being averaged with the electric minimization.
- The dipole strengths are always averaged between iterations.
- Optionally, the quadrupole strengths may be:
  - left completely separate
  - averaged along the diagonal
  - averaged across the off-diagonals which result can then be used to bias the (equal) off-diagonals of the electric projection. Due to the unknown antisymmetric properties of the quadrupole tensor, the reverse cannot be done.

[Note that in the case of two or more magnetic planes, there is no rank deficiency so that all the quadrupole tensor can be calculated before the next iteration.]

6.2 Scope of Tests and Generation of Data

The equations developed in Chapter 3 are for a bounded homogeneous sphere. In order to add some reality to their testing, a small portion of a 2m diameter sphere was used:

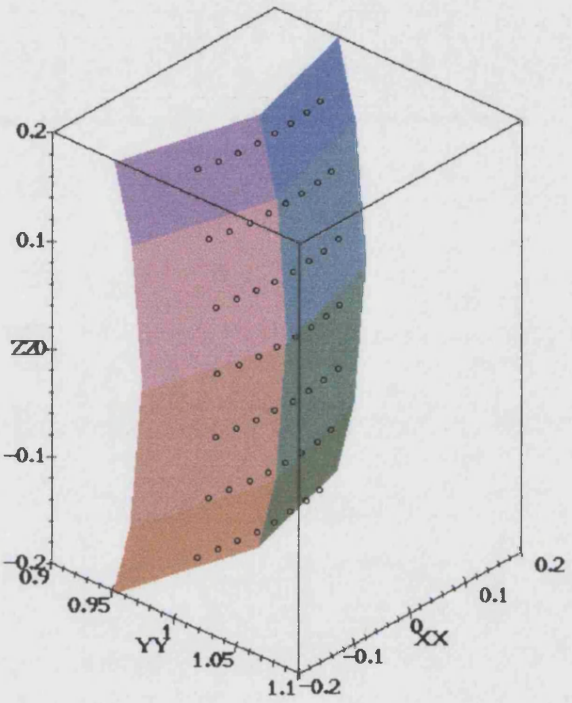


Fig 6.10 Small part of a spherical shell (patchworked for viewing) together with field measuring location points

The surface is about 36cm x 22cm and approximates the surface of a thorax. Various source positions were assumed. On the xz axis a position of (0.03,0.04) just off centre was taken as representing a point in the cardiac region. Various y co-ordinate depths were used but a shallow source was taken as about 5cm (y=0.948). Depth sensitivity testing was also carried out with depths ranging up to 23cm as representing a posterior thorax view of the cardiac region. 56 field sensor locations were assumed in a 7x8 array. In reality there would be more electrodes and a mixture of planar and axial gradiometers or magnetometers. However, to test the equations, it was assumed that the most data available would be 1 set of 56 electric potential measurements and two planes (one planar

x or z, and one axial y) of 56 unipolar magnetometer measurements. (Note that conventionally the x co-ordinate is axial, not the y co-ordinate).

In designing tests on the equations, there are a large number of possible permutations to examine:

- variations in signal strength
- dipole strength to quadrupole strength
- quadrupole anti-symmetry, dipole orientation
- signal to noise ratio
- dipole and quadrupole location, degree of separation, direction of separation, depth
- degree of data over-specification

Also the inverse model combinations and settings discussed above can be varied. Of particular interest is the impact of noise on the behaviour of the equations and, as this involves multiple statistical runs of the model, testing was necessarily limited. An initial examination of the model behaviour was conducted to determine what type of data the model was most sensitive to and where it had the greatest difficulty in predicting source locations accurately. This is not reported here but based on this preliminary work the following conclusions were reached:

- Separate locations of dipole and quadrupole sources are more difficult to find than coincident locations particularly if close to each other. A similar conclusion was reached by [Nenonen et al:1992] in the case of two dipoles.
- Strong quadrupole sources are easier to locate than weak quadrupole sources.
- Convergence is less strong for deeper sources .
- If the number of field measurement points starts to approach the number of degrees of freedom of the model, then convergence rapidly deteriorates.

These preliminary conclusions were used to generate data sets for model testing.

6.2.1 Signal Strength

From the table at the start of this section and in Chapter 1, it is possible to derive dipole to quadrupole amplitude ratios. There seems to be a wide range reported but appears to be in the region of 20:1 to 40:1. Absolute dipole strengths for this Thesis are arbitrary so for

all cases the dipole was set at: D_x=0.1, D_y=0.2, D_z=0.3 for the generation of data.
Different dipole orientations were not tested.
The quadrupole was given an anti-symmetric part and was based on varying proportions of:

$$\begin{bmatrix} Q_{xx} & Q_{yx} & Q_{zx} \\ Q_{xy} & Q_{yy} & Q_{zy} \\ Q_{xz} & Q_{yz} & Q_{zz} \end{bmatrix} = \begin{bmatrix} 0.01 & 0.02 & 0.03 \\ 0.03 & 0.03 & 0.02 \\ 0.04 & 0.01 & -0.04 \end{bmatrix}$$

This gives the dipole to quadrupole amplitude ratio a value of 4.5 and a power ratio of 13.17dB. Only in one test were different quadrupole coefficients used when all coefficients had their signs reversed.
Sets of data were generated with successively weaker quadrupoles. The following dipole to quadrupole ratios were used:

Amplitude Ratio	Power Ratio dB
9	19
22.5	27.1
45	33.1
99	40
225	47
450	53

The tests for sensitivity to source depth were carried out with a weak quadrupole strength (40dB dipole relative to quadrupole) on the basis that if this works then this should give encouragement to proceed with further investigation. The sensitivity to relative strength (dipole to quadrupole) was carried out at fixed depth (shallow source 5.5cms).

6.2.2 Noise Addition

Signal noise is a particular problem for the very sensitive magnetic field measurements carried out on the dc SQUID (Superconducting Quantum Interference Device). Noise is less of a problem for the stronger electric potential measurements but is still present. Various authors have studied noise levels. [Nenonen et al:1991] reported that a single

channel SQUID gradiometer has an average noise level of about 3 fT/√Hz. With a signal peak of 40 – 130 pT this corresponds to 2 – 3%. In subsequent tests they used rms noise levels of 50fT for the magnetic data and 2μv for the electric potential data. They noted that multi-channel SQUIDS had a sensitivity of 5 fT/√Hz. [Trahms et al:1996] reported their instrument noise was 9fT/√Hz and 0.2μv. In computer simulations [Einola et al:1993] chose 2 rms noise levels at 20fT and 100fT. They defined SNR (signal to noise ratio) as:

$$20 \log_{10} \left(\frac{\text{difference of max and min of field}}{\text{2x rms value of noise}} \right) \text{ dB}$$

Their average SNR ranged from 15 to 37 dB. This definition was adopted for this Thesis and was only found to be a small problem when generating data for significantly different depth sources since the difference between the peak values of the magnetic field narrowed.

Based on the above, Gaussian white noise was generated for the data at a high 15dB SNR level and a low 30dB SNR level. Each noise test consisted of 50 data sets and any non-convergent results were eliminated. On average each set gave 40 to 50 good results except at the extremes (eg high noise and weakest quadrupole and/or deepest sources) where it was 30 to 40. When the number of good results per set dropped below this, either the convergence control needed adjusting or the model was not able to produce a consistent enough answer to be of any use.

6.2.3 Source Location

Arrythmogenic regions can occur anywhere in the myocardium. The anterior epicardium could be as close as 5cm from a particular measuring coil and this was therefore taken as the distance to a shallow source. The distance to the posterior epicardium from the anterior thorax is typically 15cm allowing for some 3cm clearance for the measuring coil. Deeper sources could be defined when measurements are taken from the posterior thorax. If, as postulated here, the arrythmogenic region is the cause of a quadrupolar disturbance whilst the dipolar region has its electrical centre elsewhere, then a difficult localization would consist of separating the two if they are close. Possibly even more difficult would be the case where the weak quadrupole source is behind and shielded by a much stronger dipole source. This was the assumed case for testing the model in the presence of signal

noise. The shallow dipole source co-ordinates were (0.03,0.948,0.04) and the quadrupole source was (0.03,0.943,0.04) some 5mm directly underneath. Depth testing was done in 3cm steps but still maintaining the same 5mm distance between the two sources.
Best localization accuracies of 1 – 2 cm are reported in the literature [Nenonen:1992] [Moshage et al:1995] with a considerable scatter (standard deviations of a similar magnitude) due to noise and a marked deterioration due to source depth. An order of magnitude improvement on this should be the target for this theoretical model since other sources of error (for example: internal boundary effects, curvature errors, sensor location errors) will contribute to the overall error when real data are analysed.

6.3 Results

Before describing the test results, the performance of the different inverse models with clean data is described. In other words, for a range of source assumptions, how well do the models recover the true source position co-ordinates.

The separate dipole and quadrupole inverse models for locating a single point dipole and for locating a single point quadrupole were all able to converge rapidly on the original location which was used to generate the data, whether the electric potential or magnetic field versions were used, or any combination of them. These results are not reported here as they are of no particular interest. Neither were these models tested for their behaviour when noise was present in the signal, since the testing of two point sources was of more interest.

The combined dipole and quadrupole point source models had two versions: coincident sources and separate sources. As mentioned above, the coincident source model was thought to be of lesser importance than separate sources, so only the latter was used and only its performance is reported here.

Two sets of data were used: shallow sources and deep sources. The forward models were run producing true data for a dipole and a weak quadrupole, relative mean signal strength of 40dB:

Shallow source:

Dipole co-ordinates	x=0.03	y=0.948	z=0.04
Quadrupole co-ordinates	x=0.03	y=0.943	z=0.04

Deep source:

Dipole co-ordinates	x=0.03	y=0.768	z=0.04
Quadrupole co-ordinates	x=0.03	y=0.763	z=0.04

Source strength (40dB Dipole to Quadrupole):

Dipole strengths	D_x=0.1	D_y=0.2	D_z=0.3
Quadrupole strengths - magnetic	Q_{xx}=0.0004545	Q_{yx}=0.0009091	Q_{zx}=0.0013636
	Q_{xy}=0.0013636	Q_{yy}=0.0013636	Q_{zy}=0.0009091
	Q_{xz}=0.0018181	Q_{yz}=0.0004545	Q_{zz}=-0.0018181
Quadrupole strengths - electric	Q_{xx}=0.0004545	Q_{yx}=0.0011364	Q_{zx}=0.001591
		Q_{yy}=0.0013636	Q_{zy}=0.0006818

For all test runs, the inverse model sometimes would reach a local minimum and place the dipole on the wrong side of the quadrupole. These are labelled 'LM'. Where this happened, some test runs were conducted where an arbitrary 0.02 was added to the y co-ordinate after steady state had been reached and the inverse was then continued until a new alternative or global minimum was reached (labelled 'GM'). This is not satisfactory when processing unknown patient data. However, each time the error squared was lower for the 'GM'. In searching for a global minimum in unknown data, it can only be hoped that close to this minimum there will exist very few, if any, local minima. Local minima can sometimes be avoided if the starting positions are altered, as the dipole and quadrupole have differing convergence rates. The switchover point is quite sharp. A number of starting points would have to be used with unknown data and small disturbances made after a minimum reached in order to determine if the true global minimum had been found.

For inverses where only one magnetic plane is used and no electric potential data, the problem arises as to what rank to use for the linear variables. As discussed above, the infinite medium data is rank deficient but the volume current data, although a minor contributor to the magnetic field, is not rank deficient. Reducing the rank from 12 (full rank) to 10 has different effects. It certainly improves convergence but at the expense of full information on the linear (source strength) variables.

The performance results are tabulated in Appendix 4 Tables A4-3 to A4-10. Note that in arriving at these answers, a large range of convergence factors was used. Often the initial 10 or 20 iterations had to be conducted with a low rate of convergence (vj =10000) followed by successive order of magnitude reductions to the point where instability became a problem. The deeper sources could tolerate much higher convergence rates (10^2 to 10^{-1}) for the magnetic cases due to the relatively stronger volume current (full rank) data.

From these tables, a number of conclusions can be reached:

1. The electric potential inverse can find the quadrupole location for both source depths but is unable to place the dipole at its shallower position if the starting point is too close to the quadrupole. A small disturbance, after the local minimum was reached, caused the model very quickly to move to the correct global minimum.
2. Single plane magnetic field inverses for the shallow sources seemed to have a rank deficiency problem. Reducing the rank to 10 and allowing the model to choose which column vectors to include (and note that this choice could vary from one iteration to the next) gives a more reliable convergence. Later tests with a single magnetic plane are carried out but only with the B_y plane.
3. Combining 2 or 3 sets of data and doing a parallel inverse with inter-iteration unification gives a more reliable convergence.
4. The magnitude of the error squared figure is not necessarily a good guide to the best fit location. The linear variables, particularly where there is a tendency to rank deficiency, will sometimes cause high error figures without seemingly influencing the non-linear variables too greatly.
5. The performance of the model with deep sources looks similar to shallow sources. Convergence factors were very much smaller (ie faster convergence rates) although a larger number of iterations was usually required. This may foreshadow problems with uncertain data.
6. Starting positions were varied as, generally, the dipole convergence rate seemed to be slower than the quadrupole. If the quadrupole found its true location quickly, it would tend to stay there. As mentioned above, if the dipole was on the wrong side of the quadrupole, it would tend to stick there. The model never

seems to allow the two sources to occupy the same position. (In other tests, not reported here, data was run in these separate location models where the true positions did coincide. This forces a singularity due to the non-linear rank being 3 rather than 6 built into the model. The model eventually crashed)

7. The model allows the magnetic quadrupole strengths 9 degrees of freedom which is mathematically incorrect. It can at most be 8 for a symmetric plus anti-symmetric tensor. Since the original data was correct, it was left this way to see what would result. Again, in other tests not reported here, forcing $Q_{xx} + Q_{yy} + Q_{zz} = 0$ had very little effect. The electric quadrupole strengths were forced into their symmetric 5 degrees of freedom.

6.3.1 Location of a Dipole

A set of tests was carried out using the dipole only inverse model to determine the effect on the dipole location of including a hidden quadrupole of varying strengths. The purpose of this test was to determine the extent that quadrupolar activity could distort the position of a dipole if only a dipole model was being used for the inverse. What is the order of magnitude of dipole movement away from its true position and what, if any, particular direction is favoured? Localizations of arrythmogenic sites in patients with known myocardial infarctions has not produced very encouraging results particularly where a single current dipole is used. [Moshage and Achenbach:1995] report accuracies of around 1cm and [Weismuller:1995] reported best accuracies of 2-3cm although it is unclear whether this was a single dipole model. The full results, of the effect of an undetected varying strength quadrupole on a shallow dipole source and a deep one, are shown in Appendix 5. The diagrams show also the direction of movement of the dipole. The two distance graphs for the shallow and deep sources are presented here:

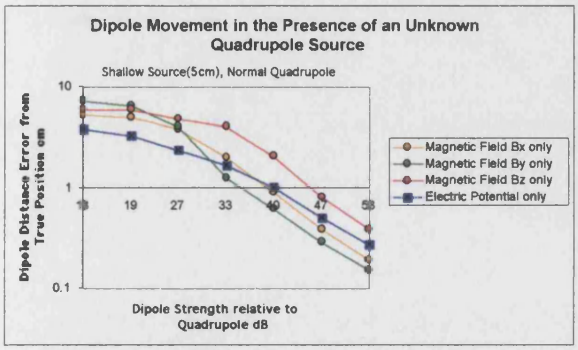


Fig 6.11 Effect on the location of a dipole when a coincident quadrupole term of varying strengths is present. Shallow source.

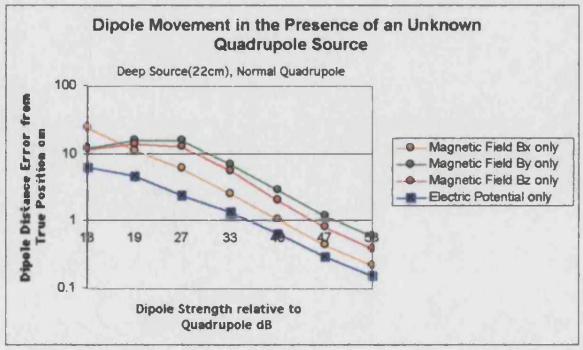


Fig 6.12 Effect on the location of a dipole when a coincident quadrupole term of varying strengths is present. Deep source.

26/7/99

Throughout the tests only one quadrupole structure was used although the strength was varied. Another data set was run for the shallow source case where the sign of the quadrupole components was reversed to see if this had any effect on dipole movement:

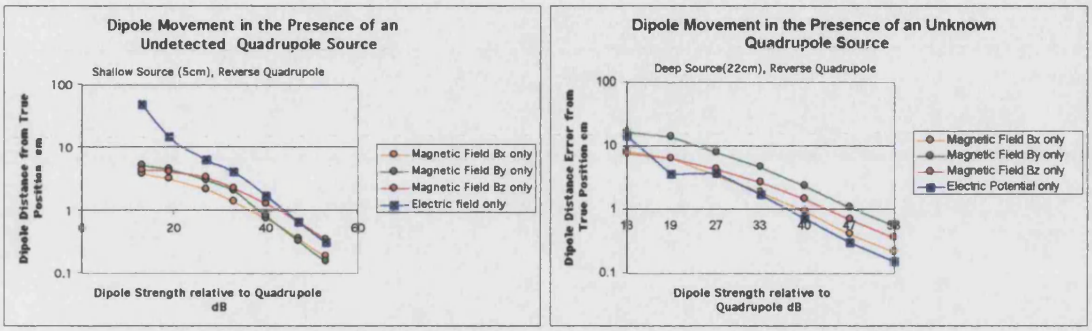


Fig 6.13 Effect on the location of a dipole when a coincident quadrupole term of varying strengths is present. Reversed quadrupole strengths. Shallow (left diag.) and deep source (right diag.).

The following conclusions can be drawn from the above and the details in Appendix 5:

1. Except for the strongest quadrupole, there appears to be a near log-linear relationship between distance error and strength (although SNR is also a log measure). The order of magnitude of the movement, for a relative dipole strength of 40dB to 47dB, is between 0.5cm and 2cm and is similar in magnitude to reported errors (also see conclusions). Note that this is not any kind of indication that quadrupolar activity is the cause of the above reported error as it might not even be present and also there are many other contributory factors.
2. It is of interest that the movement in the electrical inverse tends to be opposite to that of the magnetic inverse (which is an average of all 3 planes) and not too dissimilar in magnitude. This implies that a unified inverse model locating a solitary dipole may not be greatly affected by the presence of an unknown weak quadrupole.
3. The deeper source is slightly more sensitive to the presence of a quadrupole although the slope of the log-linear curve is very similar.
4. The different magnetic planes have different sensitivities. Of particular note is the axial plane B_y (the depth) which goes from least sensitive at the shallow depth to most sensitive at deeper positions.

5. Reversing the quadrupole polarities does not alter the overall error but more or less reverses the direction of movement.

The general conclusion is that a pure dipole model will not be accurate enough in locating cardiac current sources if a weak quadrupole source is also present. A unified electric and magnetic model may partly cancel out the error caused by the quadrupole. Deeper sources seem to be more prone to this type of error.

6.3.2 The Effect of Noise on the Location of a Point Quadrupole Source

The sets of data generated for the performance tests were contaminated with gaussian white noise at two SNR levels as defined above: 15dB and 30dB. 50 sets of data were created for each measurement profile so that a mean and standard deviation could be computed. Ideally a larger number of sets should used for each point for statistical significance but, since there were a large number of combinations, it did not prove practical to do this.

The dipole to quadrupole relative strength was kept as a parameter and different measurement sets were combined: electric, magnetic B_x, electric + magnetic B_x, electric + magnetic B_x + magnetic B_y. Taking into account the two noise levels and that some sets were run twice to obtain the global minimum, this test involved over 4000 inverses with an average 150 iterations per inverse. An iteration varied between a minimum of 1/2 sec to approximately 3 sec depending on the number of magnetic planes. Due to the near rank deficiency of the y magnetic plane, this was not tested on its own. The linear rank of all other tests involving magnetic fields was kept at 12. When the y plane is run in parallel with another magnetic plane, near rank deficiency is not a problem as only the non-rank deficient variables are selected from each plane (see program coding for details in Appendix 8).

Only the shallow source location data was tested and only the case where the dipole was located 5mm directly above the quadrupole. A separate set of tests examining the effect of depth with noisy data follows this section. Figs 6.14 to 6.25 below show the overall accuracy of the mean positions with respect to relative signal strength. Details of the movements and standard deviation diagrams are given in Appendices 6a to 6e.

The electric potential inverse had a false local minimum and therefore two sets were run:

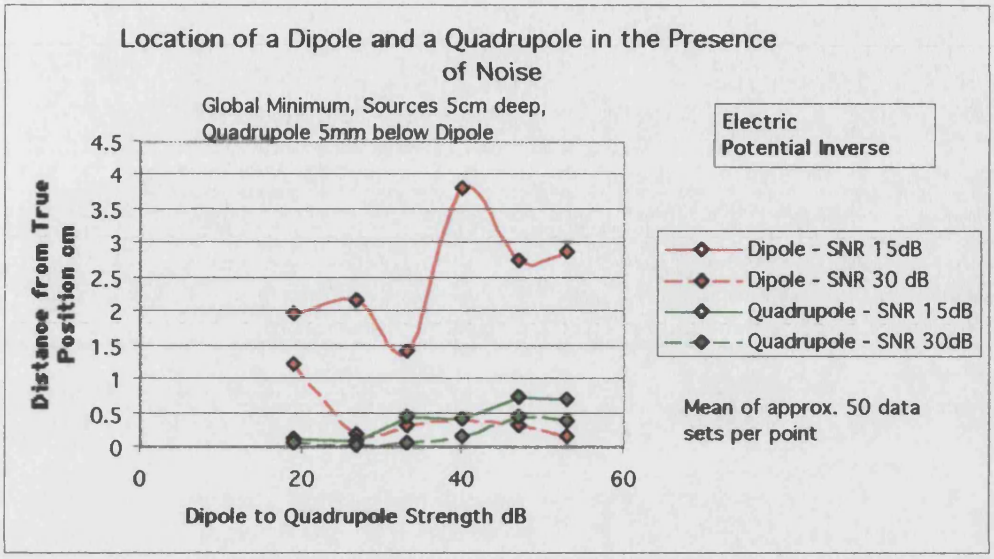


Fig 6.14 Error in the location of a dipole and a quadrupole of varying strengths due to white noise at 15dB and 30dB SNR. Electric potential inverse only, shallow source.

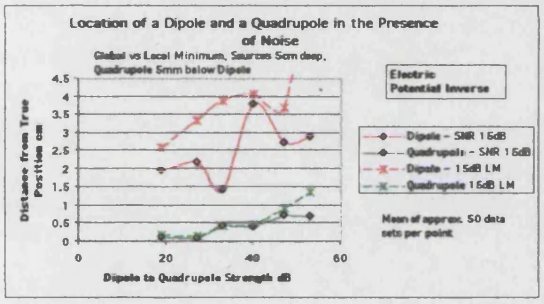


Fig 6.15 Error in the location of a dipole and a quadrupole of varying strengths due to white noise at 15dB SNR. Comparison of local and global minimum.

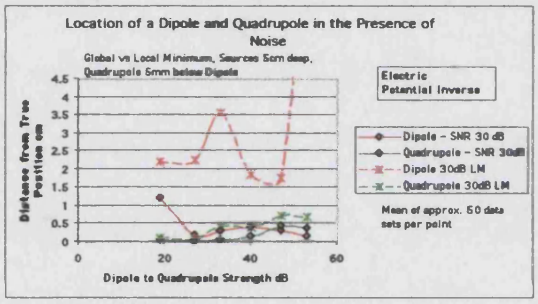


Fig 6.16 Error in the location of a dipole and a quadrupole of varying strengths due to white noise at 30dB SNR. Comparison of local and global minimum.

The conclusion from the electric inverse is that localization of the quadrupole is easier than the dipole. Even the higher noise levels (15dB) do not seem to affect the quadrupole location too greatly except in the cases of the weakest quadrupole. The local minimum mostly affects the dipole location and disturbing the model to make it converge to the global minimum causes only a small adjustment to the quadrupole location. However, the standard deviation charts (Appendix 6a) show large variations about these means for the higher noise levels. Only in the case of the strongest quadrupole is one standard deviation within ± 0.5cm across

all dimensions. The lower noise levels produce a better result with up to 40dB relative dipole strength having a similar standard deviation.

The single plane magnetic field inverse also had a local minimum requiring two sets:

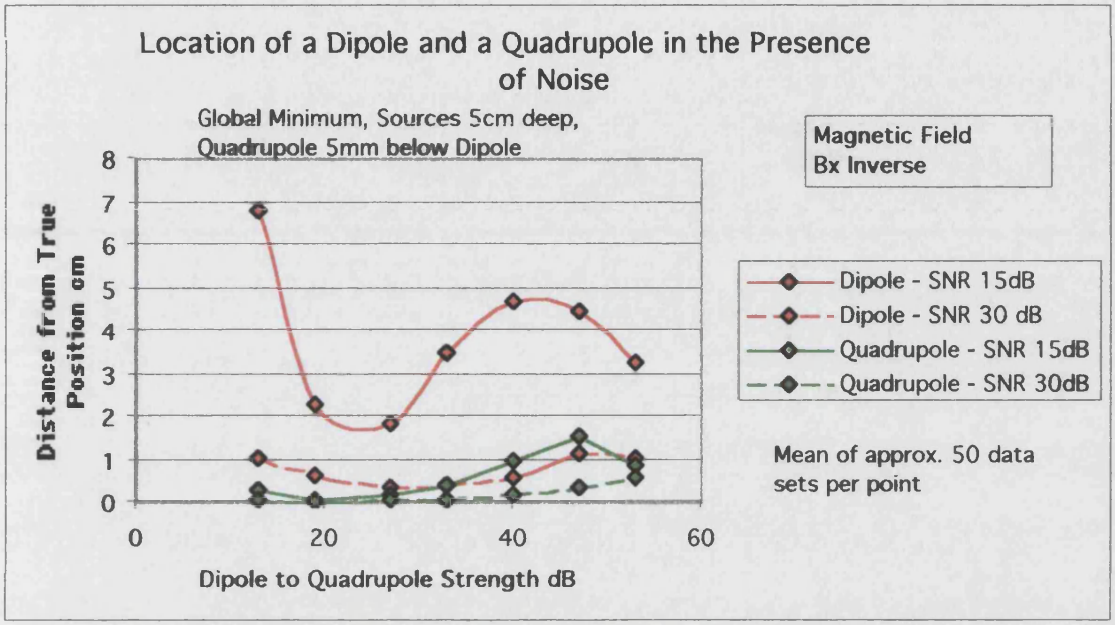


Fig 6.17 Error in the location of a dipole and a quadrupole of varying strengths due to white noise at 15dB and 30dB SNR. Magnetic field Bx inverse only, shallow source.

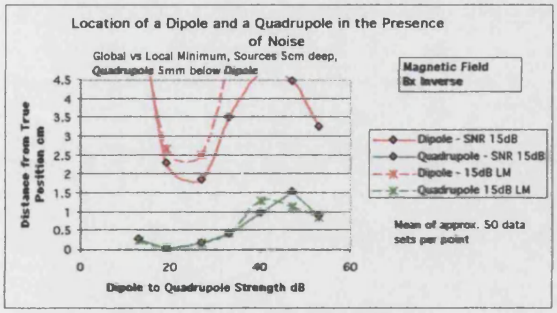


Fig 6.18 Error in the location of a dipole and a quadrupole of varying strengths due to white noise at 15dB SNR. Comparison of local and global minimum. Magnetic field Bx.

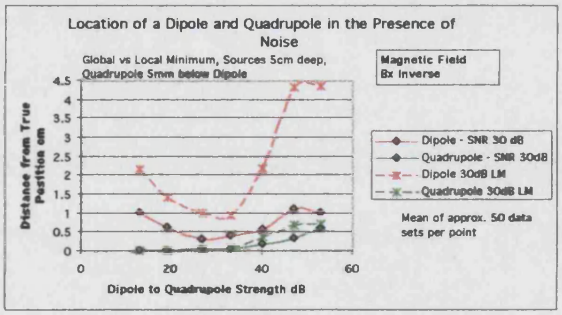


Fig 6.19 Error in the location of a dipole and a quadrupole of varying strengths due to white noise at 30dB SNR. Comparison of local and global minimum. Magnetic field Bx.

The conclusion from the single magnetic B_x inverse is that localization performance is about the same as the electric potential inverse with perhaps a slightly poorer result at the higher noise level with the weakest quadrupole source (Note that the graph scales have all been adjusted in order to present the data more evenly). The standard deviations in Appendix 6b for this case

show a slight improvement over the electric case for the 15dB noise level (2 points within ± 0.5cm across all dimensions) although the lower noise level seems to be about the same. The next set of inverses combined the electric and magnetic B_x plane:

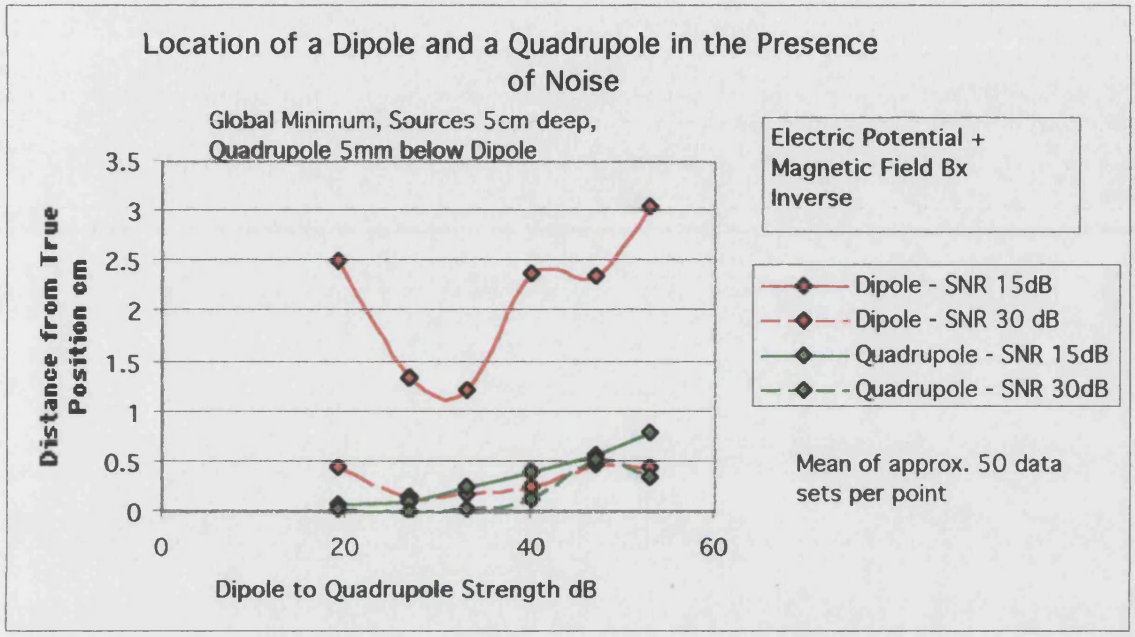


Fig 6.20 Error in the location of a dipole and a quadrupole of varying strengths due to white noise at 15dB and 30dB SNR. Electric potential and Magnetic field Bx inverse, shallow source.

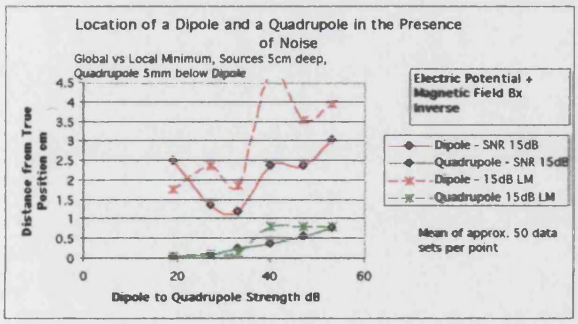


Fig 6.21 Error in the location of a dipole and a quadrupole of varying strengths due to white noise at 15dB SNR. *Comparison of local and global* minimum. Electric potential and magnetic field Bx.

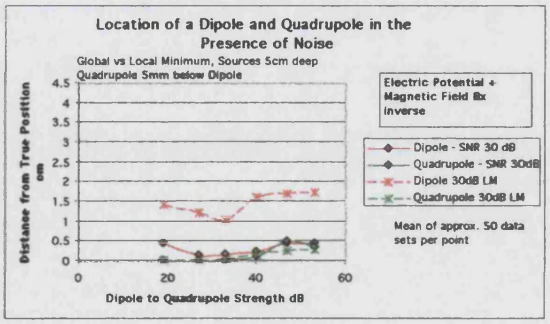


Fig 6.22 Error in the location of a dipole and a quadrupole of varying strengths due to white noise at 30dB SNR. *Comparison of local and global* minimum. Electric potential and magnetic field Bx.

The unified case produces a better result with only 1 to 2 global minimum points 0.5cm or more away from the true position and many of the quadrupole points within 1 to 2mm. The one standard deviation results in Appendix 6c also show an improvement. The noisier data has 2 to 3 points within ± 0.5cm across all dimensions and up to 4 points for the 30dB data. The 19dB to

33dB dipole relative strength data at the 30dB noise level has one standard deviation within ± 1mm to ± 2mm.

The final set of data looked at adding one additional magnetic field plane, the B_y (depth) plane. Only the global minimum case is presented:

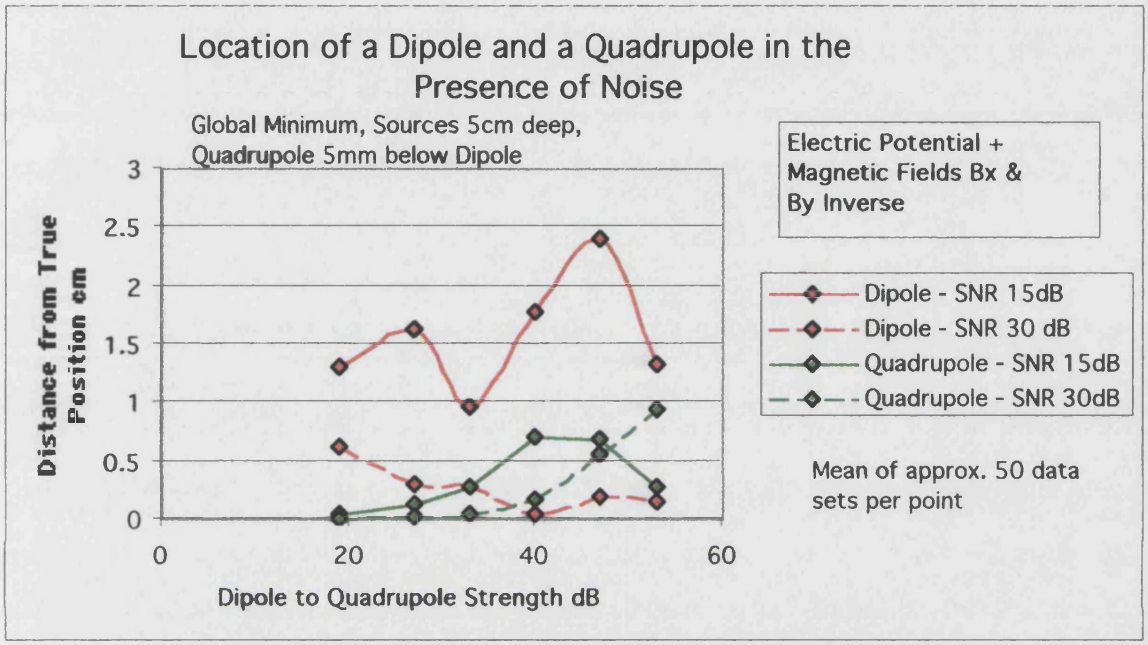


Fig 6.23 Error in the location of a dipole and a quadrupole of varying strengths due to white noise at 15dB and 30dB SNR. Electric potential and magnetic fields Bx and By, shallow source.

The slightly surprising result is that the quadrupole performance seems to deteriorate a little but the dipole result improves. The quadrupole standard deviations shown in Appendix 6d for this case also show a slight deterioration but are still better than either the electric potential or the magnetic field on its own.

Comparing the detail charts in Appendices 6a-d, the combined electrical and B_x case yields only a small improvement in quadrupole location (the x co-ordinate) and no improvement in the dipole location. However, the standard deviation of the quadrupole mean location is improved (again the x co-ordinate but particularly the z co-ordinate).

Appendix 6e shows the B_y plane on its own for the 15dB case which appears to be a much better dipole location than B_x but only a small change to the quadrupole position. Appendices 6e and 6f compare the standard deviation charts for the quadrupole 15dB and 30dB global minimum

cases but puts the different combinations of data on the same chart and uses the same scale for ease of comparison. Comparing the B_y to B_z inverses shows that in the 15dB case there is a better z co-ordinate resolution but yet when B_y is added to the electrical and B_x inverse the result is no better and for the z co-ordinate is worse. The 30dB cases show very little change.
This seems to indicate directional sensitivities to the different measurement types. A decision would have to be made as to whether a dipole or quadrupole is more important to locate.
Adding an additional magnetic plane seems to add rather little new information about location. The particular direction of movement is probably related to the individual mixture of multipole components used but there is some compensation taking place in the unified model. If only a quadrupole is sought, the addition of an extra magnetic plane for a shallow source seems to have limited benefit but considerable benefit if a dipole is sought when the signal is noisy.
There are several other combinations that could be tried, such as two magnetic planes alone and perhaps combinations where there are different numbers of measurements for the different data sets. Also the effect of changing the mixtures of strengths and directions of the multpole components could be investigated. None of these were conducted in sufficient detail to be worth reporting but instead an investigation was conducted into localization at different source depths which is the next most important issue after noise.

6.3.3 Source Depth and the Effect of Noise on the Location of a Point Quadrupole Source

Adding another parameter to the tests called for curtailment of some of the other parameters since the extra evaluation time would not justify the small gain in understanding. The depth tests were restricted to one dipole to quadrupole relative strength and one SNR.

From the previous section, it appeared that 40dB dipole relative strength was a point where the results began to deteriorate badly and certainly this was the case for the dipole. This was the level chosen for the depth sensitivity tests. In the above description on assumptions, this may be an unduly pessimistic view of the real world (ie too weak a quadrupole). In the absence of reported data and keeping in mind that the new studies being carried out, described in the introduction, may yield better information on the characteristics of signals from arrythmogenic sites, it was felt better to take this more pessimistic view.

It was also felt that, with improving equipment and technology, the lower noise levels (ie 30dB SNR) are likely to be more realistic so that only this level was tested. Other parameters were kept the same, namely: dipole to quadrupole distance at 5mm and the relative positions with dipole situated above.

Depths were selected at 3cm intervals starting with the shallow position used above in Section 6.3.2. The method chosen was merely to reduce the y co-ordinate by this amount keeping the xz co-ordinates the same, rather than try to move towards the sphere centre along the radius to the shallow point. As with the above, each point was the result of 50 sets of data with non-convergent sets being eliminated. This resulted in between 40 to 50 successfully converged results. As a side note, it was found that many sets that were eliminated because the error squared had not settled or was unrealistically large, were not noticeably different in their location results. Large error squared numbers were often due to the linear variables with apparently only small changes to source location.

Four tests were carried out with the objective of trying to improve the resolution at successively deeper source points. The following is the overall result in distance from the true positions:

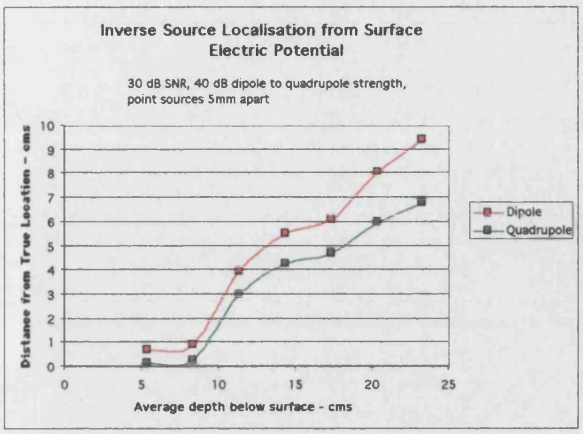


Fig 6.24 Electric potential inverse. Error in the location of a dipole and a quadrupole (40dB rel. strength, 30dB SNR) vs source depth

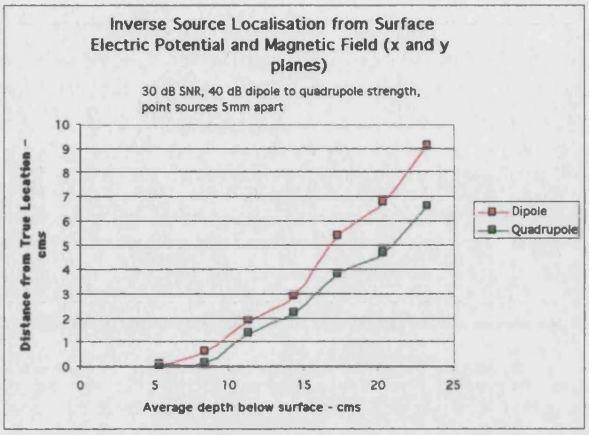


Fig 6.25 Electric potential and magnetic fields Bx and By inverse. Error in the location of a dipole and a quadrupole (40dB rel. strength, 30dB SNR) vs source depth

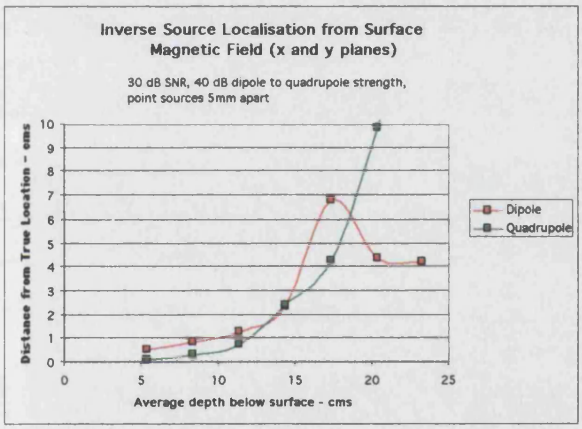


Fig 6.26 Magnetic fields Bx and By inverse. Error in the location of a dipole and a quadrupole (40dB rel. strength, 30dB SNR) vs source depth

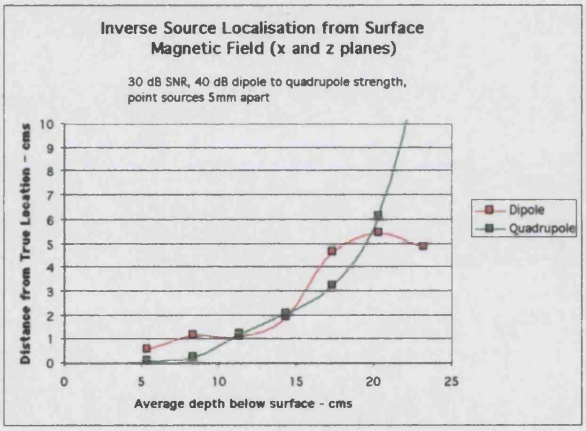


Fig 6.27 Magnetic fields Bx and Bz inverse. Error in the location of a dipole and a quadrupole (40dB rel. strength, 30dB SNR) vs source depth

These graphs show that the performance of the model at depths much over 8cm is poor. The lower two graphs were drawn in an attempt to see if the y co-ordinate had some benefit over the x and z combination. They indicate a very small advantage at the 11cm depth.

In Appendix 7a are the sets of standard deviation graphs for both the quadrupole and dipole. These show that the confidence limits deteriorate very rapidly with depth. Another feature of

these graphs illustrates that the model is unable to resolve the depth of the source (the y co-ordinate).

It should be noted that the graphs with mixed electric and magnetic data can be altered slightly by different convergence control profiles. This has already been indicated in the section above on unification where the 'centre of gravity' of the differing individual data sets could perhaps be influenced by this convergence setting.

One of many questions, hinted at earlier, which can be posed is the effect the depth (y co-ordinate) could have on the ability of the model to resolve the source depth. Two more sets were run: the magnetic field B_y profile, and this profile combined with the electric field. From preliminary trials of the model it was apparent that the B_y inverse was closer to linear rank deficiency than the other planes. Therefore, these two sets were run with the B_y rank set to 10. The model chooses which matrix column vector to include in the each iteration and almost without exception it picked D_y from the dipole and Q_{yy} from the quadrupole as the most dependent and therefore the ones to be excluded. From the preliminary trials described above, this was expected. The following resulted:

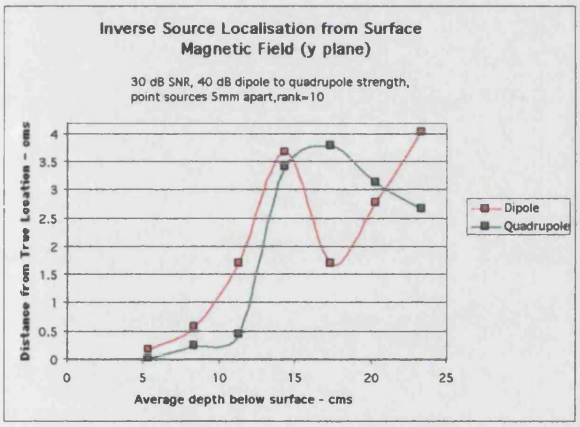


Fig 6.28 Magnetic field By inverse. Error in the location of a dipole and a quadrupole (40dB rel. strength, 30dB SNR) vs source depth. Rank=10

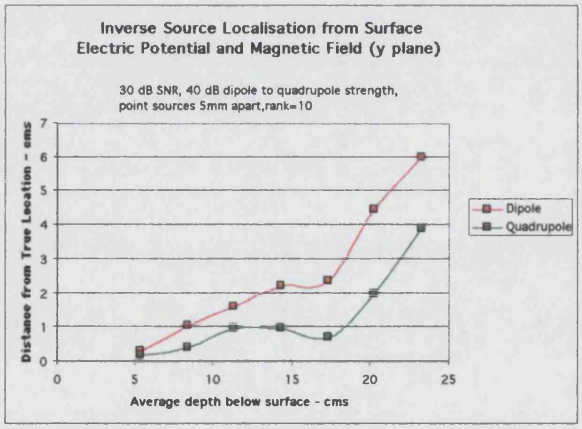


Fig 6.29 Electric poential and Magnetic field By inverse. Error in the location of a dipole and a quadrupole (40dB rel. strength, 30dB SNR) vs source depth. Rank=10

Although the B_y plane on its own seems to be only a small improvement (particularly the 12cm depth) one interesting feature which can be seen in the confidence limit plots in Appendix 7a and b is that the electric data tends to undershoot the y co-ordinate whereas the B_y plane data seems to overshoot. Combining them produces an average which is much closer to the true y co-ordinate. Also it can be seen that although the means are improved, the confidence limits are not. The reason for this lies mostly in which results were selected and which were rejected

because of non-convergence. A much tighter definition based on error squared magnitude and error squared inter-iteration movement of the separate electric potential and magnetic field errors would give improved confidence limits.

A general conclusion is that there is some evidence that the electric inverse error counteracts the magnetic field error. The tests on dipole movement in the presence of an unknown quadrupole source indicate this is the case for the dipole movement and this study indicates something similar, although much less certain for the quadrupole source. One could therefore question the wisdom of trying to combine the minimizations where so many factors could influence at what point the convergence will end up, as opposed to keeping the inverses separate and then doing a weighted average of co-ordinate results based on some view of the confidence bands of each data set.

6.3.4 Model Convergence Sensitivity

In order to examine the ability of the combined dipole and quadrupole, (electric and magnetic) models to locate a weak quadrupole more strongly than a dipole, a small set of step perturbations from the true positions was run with a single iteration to see the magnitude and direction of the response of the variables close to convergence. Each co-ordinate of each point source was altered from its original position in turn by differing amounts.

The test was carried out by taking data for the 40dB relative strength dipole without noise and the true position depth was 17cm. The size and direction of the step towards (or possibly even away from) the true position could then be compared for each measurement plane for a range of initial displacements. Also the effect of the convergence control is compared by using two convergence settings. Figs 6.30 to 6.33 show the movements resulting from the single iteration:

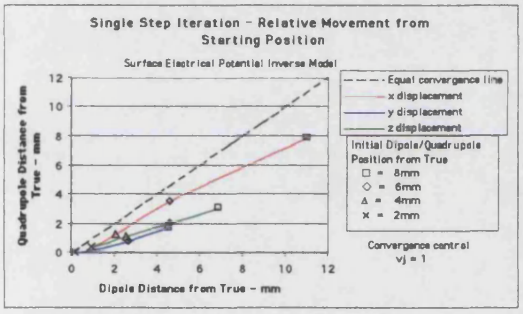


Fig 6.30 Model sensitivity near convergence. Electric potential inverse single step iteration. Movement of dipole and a quadrupole (40dB rel. strength) towards true position. vj=1

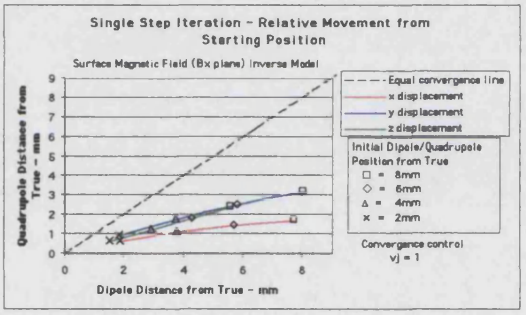


Fig 6.31 Model sensitivity near convergence. Magnetic field Bx inverse single step iteration. Movement of dipole and a quadrupole (40dB rel. strength) towards true position. vj=1

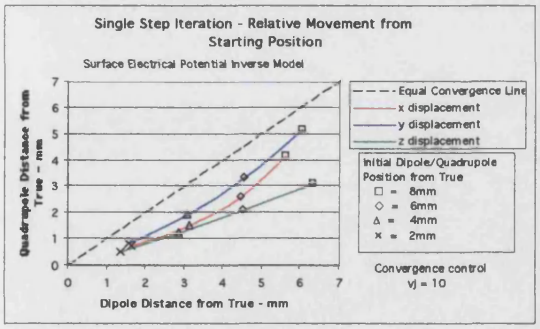


Fig 6.32 Model sensitivity near convergence. Electric potential inverse single step iteration. Movement of dipole and a quadrupole (40dB rel. strength) towards true position. vj=10

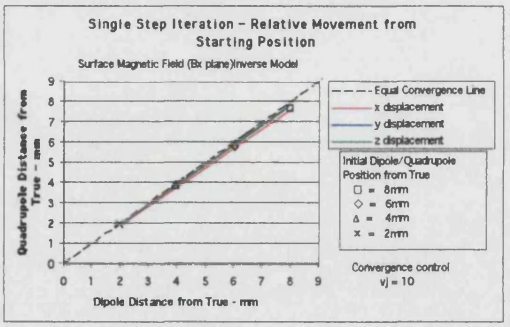


Fig 6.33 Model sensitivity near convergence. Magnetic field Bx inverse single step iteration. Movement of dipole and a quadrupole (40dB rel. strength) towards true position. vj=10

These diagrams show for the electric potential model and, separately, for the B_x magnetic field model, the response of the dipole compared to the response of the quadrupole. Any points on the dotted diagonal indicate that the movement of each point source was the same. In all cases except one, the lines are below the diagonal indicating that the quadrupole is making a larger step towards its true position than the dipole. The exception is the lower convergence rate (vj=10) for the magnetic model where the movements from the starting positions for both the dipole and quadrupole are extremely small. For the other cases the dipole is moving towards its true position but, when the detailed data are examined, exhibits some lateral or rotational movement as well. In other words all the co-ordinates (x,y and z) are changed even though two of the three were correct before the step.

Of more significance is the behaviour of the unified model. In Figs 6.34 and 6.35 the electric potential inverse model behaviour is compared with the x magnetic plane:

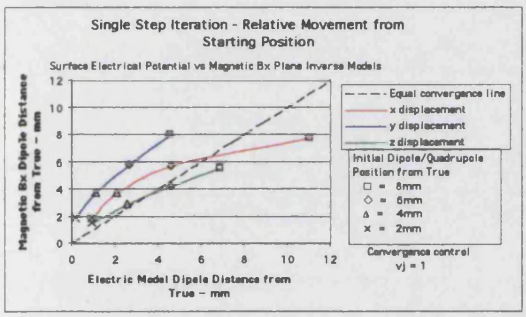


Fig 6.34 Model sensitivity near convergence. Electric potential inverse vs magnetic field inverse single step iteration. Movement of dipole towards true position. vj=1

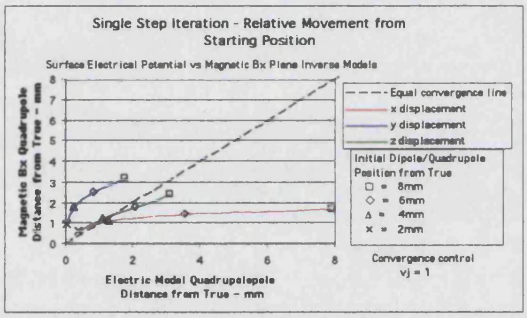


Fig 6.35 Model sensitivity near convergence. Electric potential inverse vs magnetic field inverse single step iteration. Movement of quadrupole towards true position. vj=1

In the left diagram, the dipole movements are plotted against each other and, on the right diagram, the relative quadrupole movements are shown. The general conclusion is that, close to the true locations, for the dipole there is a slight bias towards the electric inverse model but the quadrupole is more or less neutral. The exception seems to be displacements in the y co-ordinate. This is the depth co-ordinate and, as results have shown, the performance of the model deteriorates markedly with source depth. It should be noted that for all of the above studies on localization it was found best to put the electric setting to a weaker convergence than the magnetic. This would be consistent with these graphs.

The single step inverses were also carried out with displacement of the dipole only and repeated with displacement of the quadrupole only, instead of the above tests where both were changed. Therefore, in each test one source started at its correct location. These are shown below for the quadrupole only displacement:

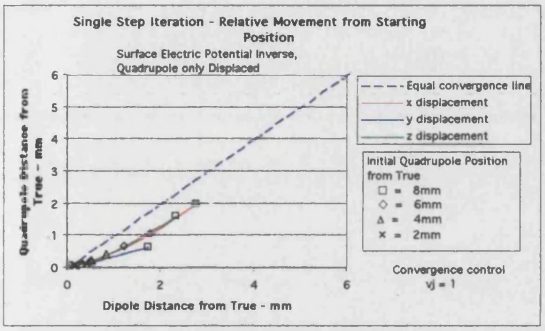


Fig 6.36 Model sensitivity near convergence. Electric potential inverse single step iteration. Movement of dipole from true and movement of displaced quadrupole towards true position. vj=1

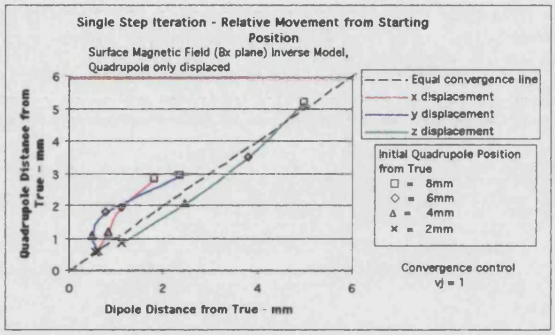


Fig 6.37 Model sensitivity near convergence. Magnetic field Bx inverse single step iteration. Movement of dipole from true and movement of displaced quadrupole towards true position. vj=1

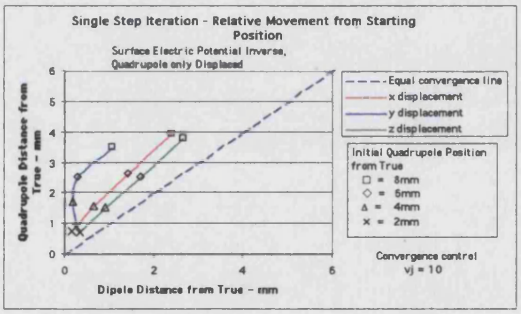


Fig 6.38 Model sensitivity near convergence. Electric potential inverse single step iteration. Movement of dipole from true and movement of displaced quadrupole towards true position. vj=10

The interpretation of these graphs, and the hypothesis put forward here, is that the inverse tries to create two multipole series that best fit the data. However, the model is constrained to eliminate all multipoles higher than the dipole in the dipole's location, and all multipoles higher than the quadrupole in the quadrupole's location. Data which does not fit, such as a displaced quadrupole, has to be distributed amongst the free parameters within these constraints. Therefore, an additional dipole element is free to appear at both locations. In one iteration, this moves the correct dipole off its true position and moves

the quadrupole to a new location. Since the quadrupole linear parameters have been given 9 degrees of freedom (which is theoretically incorrect but leads to a better inverse), then some skewing of the quadrupole tensor takes place and the diagonal does not vanish. It is possible that this is effectively the dipole term at the quadrupole location as it is not explicitly programmed into the solution. Further work is suggested that allows such a dipole to exist and whether this leads to a better inverse solution. It would also be useful to know whether cardiac electrophysiology allows such a combination to exist at the boundary of an infarct. What is clear from these graphs is that the electric inverse moves the dipole some way from its true position, further than the quadrupole from its true position. On the other hand, the magnetic inverse moves the dipole only a small amount from its true position. A few more iterations and both arrive at their true locations. The electric inverse at a slower convergence setting behaves in the same way as the magnetic inverse (Fig 6.38).

The dipole displacement from its true position has not been greatly affected by the convergence setting but the quadrupole movement towards its true position has been. These graphs tend to support the results, namely that the quadrupole convergence near to its true position is likely to be stronger than the dipole and that residual error is preferentially distributed to the dipole. A similar exercise could be conducted with the linear variables and also with both the linear and non-linear variables with small displacements in values close to the true ones.

As a final comment on the question as to whether the magnetic data can add new information to the electric data: the questions on rank deficiency have been pointed out above but this is only in relation to the linear variables. It would seem that both sets of data are able to recover the source co-ordinates (depending on noise levels) but that there are different sensitivities in different planes.

6.4 Conclusion

Comparison of this Thesis with other studies has already been commented on in relation to generating data. In relation to the impact of noise on source localization, the exercise is equally difficult for the following reasons:

- many people have used just the current dipole model.
- some authors restrict the dipole to one axis or examine a fixed orientation.
- papers where a magnetic field is examined usually only include a magnetic dipole (the anti-symmetric quadrupole).
- the favoured model seems to be the half-space not the sphere. The sphere is more common in neuro-magnetic studies but these are nearly always based on a current dipole source.
- realistic torso modelling has received the most attention in recent years and the causes of noise related source location error are very hard to pinpoint in such models as they contain all the possible error sources in this Thesis plus a number of others.

The following review looks at some relevant work and tries to make a comparison. Unfortunately, many papers give insufficient detail of the assumptions made or incomplete (or relative) data making direct comparisons impossible. However, in magnitude and direction there is some agreement.

The work by [Cuffin:1986] was on a spherical model using a current dipole to generate a magnetic field in a 9cm radius homogeneous sphere. The source was 2cm below a flat grid of 49 measurement points. He added noise at 20% and 10% (14dB and 20dB?) and reported at the 10% level:

	error of mean	standard deviation
x co-ord	1 – 2mm	3 – 4mm
y co-ord	2 – 5mm	9 – 11mm
z (depth)	<1mm	6 – 7mm

and at the 20% level:

	error of mean	standard deviation
x co-ord	3 – 7mm	5 – 13mm
y co-ord	2 – 4mm	12 – 18mm
z (depth)	4 - 6mm	8 – 11mm

The ranges are due to two different measurement grid arrangements.

The same author in an earlier study [Cuffin:1985] had concluded that effects of noise on a current dipole were about the same when using the electric or magnetic fields.

[Buchannan:1989] examined a current dipole in a 12cm radius sphere at a depth of 5 to 7cm. He used a source strength of 10nAm and added field noise of 20fT (estimated as 20dB as this was neuro-magnetic work). He added other sources of error such as relative channel gain error and translational/rotational system placement errors. He reported mean location errors of 2 to 5mm.

[Einola et al:1993] used Boundary Element Modelling of a real torso to examine the sensitivity of different magnetic planes. His measurement grid consisted of 37 magnetometers for B_z (depth), 37 axial gradiometers for $\partial B_z / \partial x$ and 37 planar gradiometers for $\partial B_z / \partial x$ and $\partial B_z / \partial y$

The forward field was computed for a number of specific locations and a series of depth positions. 100 sets of data contaminated with 100fT of noise were then generated for the inverse model at SNRs of approximately 15dB to 40dB. Somewhat in line with this Thesis, there was a marked deterioration of accuracy between the 10 to 15cm depth levels. Mean location errors of <5mm were found at depths less than 10cm and 5 – 20mm at 10 to 15cm depths with standard deviations of ± 50% to 100% of the absolute error. Interestingly he also noted that there was slightly better accuracy if the measurement plane was curved so that sensors followed the chest contour a little better.

The best study for comparison was that carried out by [Gonelli:1991]. He used a homogeneous half space with a source depth of 6.5cm and examined the impact of noise on the magnetic field inverse (not reported but presumed to be a single plane field, the B_z depth plane). The more interesting part is that he tested combinations of different point source models: the equivalent current dipole (ECD), the equivalent magnetic dipole

(EMD, the anti-symmetric quadrupole) and the current multipole expansion (CME, the full quadrupole plus the dipole). He added noise at two levels, 15dB and 30dB, and he used a relative dipole to quadrupole strength of 30dB. The dipole and quadrupole point sources had the same location. Fitting the ECD model to an original source generated by the CME (similar to the exercise in Section 6.3), he reports errors of 5mm for the lower (30dB) noise level and 10mm for the higher level. This compares with 10 – 20mm for noise free data reported above. Exact comparisons cannot be made as he used a different quadrupole strength arrangement with a different symmetric/diagonal ratio. Fitting a CME source to a CME model:

	Gonelli	This Thesis
depth	6.5cm	5cm
location mean error:		
15dB SNR	4mm	18mm dipole 4mm quadrupole
30dB SNR	2 - 3mm	4mm dipole 1mm quadrupole

The equations developed in Chapter 3 have been tested in this chapter in a way that tries to scale the geometry to that which would be found in a realistic torso. Only the most important aspects of source arrangements were tested. Many others tests could be applied such as:

- coincident dipole and quadrupole sources
- as above but with a second nearby dipole
- dipoles located further away from the quadrupole
- sensitivity to different combinations of model parameters
- different electric and magnetic unification methods (such as constraining the number of degrees of freedom of the linear variables)
- sensitivity to different quadrupole structures

Exhaustive testing is of little value at this point because there is insufficient knowledge of the electrophysiology of ischemic regions and the nearby tissue in the myocardium to indicate what the parameters ought to be.

The unexpected conclusion from these tests is that it is easier to locate a weak quadrupole source in the presence of a realistic level of white noise than a nearby stronger dipole source. The model distributes the errors preferentially towards the lower order source. Sensitivity graphs support this result. The non-linear part of the inverse model is driven by the next order differential with respect to the source location variables and this is probably the explanation of this result. The quadrupole field profile, although smaller in magnitude, has steeper gradients (2 sets of minima and maxima as against one) than the dipole. It should be possible to prove this mathematically by examining the differentials of the equations in relation to the variable projection method.

With regards to the unification of electric and magnetic fields, there is some evidence that their respective errors are in opposition and therefore are self cancelling. The method of unification needs to be selected carefully as a combined minimization cannot by default, produce a better answer. It produces a weighted view of the two separate minimization results. Certainly in this model the weighting can be biased to favour one profile or the other. Certain degrees of freedom can be restricted to force the model into a consensus view of the magnetic and electric linear variables:

1. dipole values always the same
2. quadrupole diagonals always the same
3. average quadrupole off-diagonals always the same

Ideally all three should be applied but the test runs carried out here only applied to 1.

The mean errors reported in this Thesis are of the same magnitude found by others to the extent that any comparison is possible. Often the mean errors were fairly accurate but the standard deviations were disappointing. Until these can be narrowed, doubt will always remain about the accuracy of the localization result and localization will stay as a supportive clinical technique rather than a predictive one. The way to improve standard deviations from these tests is to reduce noise levels and to have the appropriate model (source description, rank, degrees of freedom, unification method).

7. Conclusions

The principal conclusion of this study is that it is easier to locate a weak quadrupole current source in the presence of a realistic level of noise and in the presence of a nearby stronger dipole current source than it is to locate the dipole itself. This is an unexpected and counter-intuitive finding. So what does it mean?

The original aim of this PhD study was to examine higher order terms in the multipole series representation of the cardiac bioelectric equivalent source. Such a source is capable, at least mathematically, of approximating with reasonable accuracy the electric and magnetic fields generated by the highly complex real source. Until recently almost all research work has used the lowest order term, the dipole, to represent the equivalent source. The electric field generated by a dipole falls off according to the reciprocal of the distance cubed whereas the next term, the quadrupole term, is inversely proportional to the fifth power of the distance and, even at the same equivalent source strength, which it is unlikely to have, will be considerably less significant. It was argued that including such a term was probably not worthwhile considering that the mathematics are considerably more involved and white noise in the signal would be at least as significant and mask any effects.

The development of low temperature magnetic measuring techniques (the SQUID) during the late 1970's revived interest in the quadrupole term as the representation of the magnetic dipole is contained in it and, if present in the cardiac source, would be detectable only by this technique and not by the more conventional ECG method. The magnetic signal from the heart is approximately one millionth that of the Earth's magnetic field and signal noise is a significant problem. However, instrumentation and the processing of the signal have made rapid advances.

The questions that were put forward as the starting point for this PhD were in the nature of signal processing issues:

- is it worth including any higher order terms and if so how many?
- what level of signal noise can be tolerated?
- can the error signal, calculated by subtracting the field generated by a dipole equivalent source from the total measured field, be modelled using higher order terms?

- by combining both electric potential and magnetic signals can a better error signal be generated to which higher order terms could be fitted?

A survey of the literature revealed that the mathematics of the higher terms, particularly for the magnetic field, are indeed cumbersome particularly as the effect of conductivity boundaries has a significant impact on the field generated outside the body and requires a surface integration to be carried out. For this reason much attention has been paid in recent years to realistic torso modelling using numerical methods to approximate the integration. The issue of signal noise is always present and has been addressed by many researchers. With realistic models and the mathematical assumptions needed to construct them, such research has shown the magnitude and effect of noise, but analysis of which part of this construct contributes how much to the effect, has not been conclusive. It was apparent from the outset that to answer the above questions, an analytical solution to the mathematics would have to be used and such analytical solutions are only possible with simple geometric shapes.

Stepping back from the mathematics and physics, the question arises as to what clinical purpose would be served if an answer to these questions could be provided? As mentioned in the introduction, the objective of improved clinical diagnosis is often forgotten by many researchers and it is important that the link in any piece of research is made. Cardiac arrythmias caused by ischemic heart muscle and infarcts are one of the most common forms of heart disease leading to more than 50% of all cardiovascular deaths. The ability to localize ventricular signals emanating from the regions around ischemic tissue using a rapid non-invasive method would be of great assistance in diagnosis and, if reliable and accurate enough, could avoid the need for pre-operative catheter mapping and possibly helpful in catheter ablation treatment. Localization has long been one of the goals of inverse modelling using body surface maps of the fields generated by the heart.

The electrophysiology of healthy heart tissue is well known but diseased tissue is less well understood and in particular the nature of injury currents has only recently become clearer. The higher frequency components found in low amplitude signals in the late part of the QRS cycle have attracted attention in patients with known malign arrythmias and these are attributed to re-entry excitation pathways around ischemic regions. The larger

of these pathways, of the order of 2cm, have been measured but it is believed that there are also micro re-entry pathways of as little as 2mm. Such pathways may not give rise to ectopic beats but may be detectable and localizable non-invasively when the right mathematical model, a higher order source term, is applied to reasonably noise free signals. The objectives of this study therefore changed to answer the question as whether a quadrupole current source on its own can be detected in the presence of noise and particularly in the presence of a stronger dipole source (possibly the still active healthy tissue elsewhere in the heart). It is unknown at this stage whether the best equivalent source model to locate such an injury current is a pure quadrupole source or whether it contains a dipolar element as well, but the magnitude of such currents is likely to be small compared to the main 'dipolar' wave from healthy tissue, if that is still present. The analytical model, the homogeneous bounded sphere, developed in this Thesis aimed to find out whether a weak quadrupole current can be localized and, if so, to quantify the impact of noise on the accuracy of the localization of such a source. Also it aimed to find out whether combining the electric potential data with the magnetic data can yield a better answer. This is part of the continuing debate on whether magnetic data, which is many times more expensive to acquire, does add new information to the electric potential data.

Chapter 2 showed that it is possible to model separate arbitrarily located solitary dipole and solitary quadrupole sources. A geometric interpretation of the quadrupole was demonstrated linking the geometry to the application of Maxwell's equations. Although the dipole is easier to picture, there is no physical interpretation of either of these sources although in the case of the dipole there appears to be a closer analogy to the cellular ionic mechanism.

The cumbersome mathematics led to the desirability of using closed form equations for the integral solution to the Maxwell equations. A closed form represents an entire mutlipole series for any point source arrangement arbitrarily located in a conducting volume. Such a closed form had been derived some years ago for the dipole electric potential solution in a bounded sphere. In Chapter 3, this solution was extended to the quadrupole case and also the magnetic field for a dipole and a quadrupole. The solution gave the bounded volume electric potential on the surface but only the infinite medium

magnetic field values. The solution to the surface integration was therefore derived using an open form harmonic series. This was a fairly lengthy and cumbersome solution.
As part of the original objectives of this Thesis, the differences between the error field on the surface of a sphere compared to a prolate spheroid had been included, as the thorax could perhaps be more closely modelled by such a shape. Chapter 4 showed an analogous solution for the electric potential and the magnetic field on the surface of the prolate spheroid. No closed form could be found for this shape so only an infinite series solution was developed including the correction for the surface boundary. This solution was not computer modelled but is nonetheless still of interest.
As the computer stage approached, the size and highly non-linear nature of the inverse solutions became of concern in that only one technique for the least squares minimization was widely used and this technique gave equal status to the large number of linear variables and the smaller number of non-linear variables. A different technique, the variable projection method, had been pointed out to the author and had been found to be much more efficient, although the application had been slightly different. This technique was studied and applied to the new equations. Compared to the more widely used technique, the Levenburg-Marquardt method, it was indeed found to be more efficient particularly when a modification published subsequently to the original paper was applied. The technique is described in Chapter 5 together with the results of the comparison.
The approach to computer modelling the equations is outlined in Chapter 6 but it is difficult to describe the coding in any detail. Of more interest is the way the equations were used and data selected to simulate a real world situation. A small portion of a large radius sphere was taken with a limited number of data measurement points. Calling the dimensions centimetres gives some idea of the model in relation to a thorax.
A large number of variables could be altered and questions asked but it was concluded that the most important were:

1. What is the impact on the location of a dipole if an unknown weak quadrupolar current source of varying strengths is also present?
2. Is it possible to locate two adjacent point sources, one dipolar and the other quadrupolar?

3. If so, what is the effect of different noise levels on the model's ability to locate and separate two such sources?
4. It is well known that deeper sources are more difficult to locate than shallow sources, so how well does the model perform the deeper the source?
5. What are the sensitivities of the model to the different measurement planes (ie the electric potential and the 3 magnetic planes)?
6. Are there any advantages in combining two or more measurement planes and how should they be combined in an inverse model?

As most tests included the effects of noise, a large number of trials was needed in order to measure a mean and standard deviation for each point on every graph. This necessarily limited the number of tests which could be conducted, even keeping the number of trials to 50 per point (which limits the statistical accuracy). The conclusions, where noise is included, therefore are indicative:

1. The effect of the unknown quadrupole without any signal noise present is to displace the dipole, and the amount of displacement bears a log-log relationship to the strength of the quadrupole.
   Each measurement plane has a slightly different magnitude of response. Of particular note is that the direction of movement of the dipole source in the electric inverse tends to be opposite to that of the averaged magnetic inverse response. The implication is that a unified model would give a better answer for a dipole location when unknown quadrupolar activity is present. The clinical diagnostic importance of this conclusion is unknown but probably of little significance.
2. It is possible to locate a point dipole source at depths of around 5cm and a separate point quadrupole source 5mm below it when there is realistic signal noise present:
   a) Except when the quadrupole strength drops to lower than –50dB relative to the dipole, the location accuracy of the quadrupole, depending on the level of signal noise, is up to an order of magnitude better than the dipole and in some instances higher.

b) Combining electric and magnetic inverses improves the dipole location accuracy but has only a small beneficial influence on quadrupole location accuracy.

c) The results also demonstrated the importance of ensuring that a global minimum is reached by the inverse. Local minima were always poorer than global but the dipole local minimum was particularly poor.

d) The results also include triplets of diagrams showing individual co-ordinate direction movements and the confidence limits of those movements for the quadrupole. These charts are harder to interpret but in general the unified inverses give slightly better standard deviations. There is also evidence that the different measurement planes have different directional sensitivities but the self cancellation seen in 1. above is weak to non-existent.

3. The higher level of signal noise, 15dB as against 30dB, adversely affects the accuracy of the dipole location. The quadrupole location is affected but only where it is weak compared to the dipole, –25dB or lower. Confidence limits for the quadrupole mean location are better with lower noise but the unified inverses seem better able to handle the higher noise levels.
4. Deeper sources are more difficult to locate. At about 8cm depth the inverse model accuracy and quadrupole confidence limits deteriorate. The depth co-ordinate could not be resolved properly and this appeared to be related to the solution matrix rank. As the variable projection method allows rank to be altered, a lower rank was used and the depth resolution improved. The electric inverses tended to undershoot the y co-ordinate whereas the B_y (axial) magnetic plane seemed to overshoot. Combining them produces a better average.
5. The sensitivity tests showed that the inverse models are more sensitive to quadrupole movement than dipole movement. This is consistent with the fact that the incremental next-step movement of the non-linear variables is determined in the variable projection method by the differential of the individual quadrupole functions with respect to each co-ordinate plane. In simple English, the contours of the quadrupole have sharper curves and, even though magnitudes are lower than the

dipole and noise appears to swamp the signal, there is enough information left to enable accurate inverse localization. Signal error appears to gravitate towards the dipole. It is for this reason that an attempt to model a residual error signal after the subtraction of a best fit dipole (an original objective) completely failed and was abandoned.

6. As regards the unification of the different measurement types, both the electric and magnetic data are able to recover the source co-ordinates (depending on noise levels) but they have different sensitivities to the different co-ordinate planes. Unification can produce a slight improvement in confidence limits. This conclusion was recently reached by [Malmivuo et al:1997] who said that what the Helmholtz Theorem (divergence and curl) expresses is not the independence of electric and magnetic signals but the independence of the sensitivity distributions of the flow and vortex sources. This Thesis tends to support this.

The overall conclusion is that the hoped for quantification of the effect of different parameters on dipole and quadrupole location has not been possible even with an analytical model. However, the trends and sensitivities are clear enough. Larger test runs could lead to confirmation of the trends observed in this work. However, there is enough evidence that weak signals emanating from the border zones of an infarct could be detectable and localizable using a quadrupole model either on its own or with a dipolar presence. The localization accuracy, remarkably, is not as sensitive to signal error as had been expected. Conductivity interfaces have caused many problems for other researchers but it could be asked whether they are needed in quadrupole localization? Test runs in this Thesis to demonstrate this would not have yielded any useful information due to the symmetry of the geometric model used.

This model should now be applied to real patient data. This data would have to be very specific to the conditions which could yield an answer:

- patients with a known malign arrythmia from a single (preferably posterior) infarct causing a fractionated electrogram.
- good quality filtered electric and magnetic data as specific as possible to the correct point in the cardiac cycle.

If, as a result, there is confirmation that localization with sufficient accuracy is possible, the equations should then be applied to a realistic torso model.

8. References

Arthur, R.M. and Geslowitz, D.B., Effect of inhomogeneities on the apparent location and magnitude of a cardiac current dipole source. *IEEE Trans. Biomed. Eng. 17 No 2 pp141-146 1970*

Achenbach S., Moshage W., Göhl K., Abraham-Fuchs K., Schneider S. and Bachmann K., Magnetocardiographic investigation of the origin and propagation of cardiac arrythmias. In *Biomagnetism: Clinical aspects, Hoke M., Erné S.N.,Okada Y.C. and Romani G.L., (Eds) Excerpta Medica, ICS 988, 1992*

Baule G. and McFee R., Detection of the magnetic field of the heart. *Amer. Heart J. 66 pp95-96 1963*

Baule G. and McFee R., Theory of magnetic detection of the heart's electrical activity. *J. Appl. Phys. 36 No 6 pp2066-2073 1965*

Baynham T.C. and Knisley S.B., Resistivity calculations in arterially perfused rabbit ventricles using a 4-electrode method. *IEEE Proceedings of the 19th International Conference EMBS Oct1997 Chicago, IL. USA.*

Berry, P.M. N,M space harmonics of the oblate spheroid. *Ann. N.Y.Acad. Sci. 65, 10126-1134, 1956*

Blimke J., Ivolkmcr I., Spilker M. and Myklebust J, Comparison of evoked potential source localization and functional MRI in a four-shell ellipsoidal volume conductor. *IEEE Proceedings of the 19th International Conference EMBS Chicago, IL. USA, Oct1997.*

Brody D.A., Terry F.H. and Ideker R.E., Eccentric dipole in a spherical medium: generalized expression for surface potentials. *IEEE Trans. Biomed. Eng., 20, 141, 1973*

Brody D.A., The inverse determination of simple generator configurations from equivalent dipole and multipole information. *IEEE Trans. Biomed. Eng., 15 No 2, 106-110, 1968*

Buchanan, D.S., Impact of different noise sources on dipole localization in the spherical model: a simulation. In *Advances in Biomagnetism, Williamson, S.J., Hoke, M., Stroink, G., and Kotani, M. eds, Plenum Press, New York, 1989.*

Budgett D.M., Remote sensing of the epicardium, *PhD dissertation, Imperial College of Science, Technology and Medicine, London,England. 1995*

Clerc L., Directional differences of impulse spread in trabecular muscle from mammalian heart *J. Physiol. 255, pp.335-346, 1976*

Cuffin, B.N. and Cohen, D. Magnetic fields of a dipole in special volume conductor shapes *IEEE Trans. Biomed. Eng., 24,2372-381,1977*

Cuffin,B.N., A comparison of moving dipole inverse solutions using EEG's and MEG's. *IEEE Trans. Biomed. Eng., 32,905-919,1985*

Cuffin,B.N., Effect of measurement errors and noise on MEG moving dipole inverse solutions. *IEEE Trans. Biomed. Eng., 33, pp. 854-861,1986*

Cuffin,B.N., Eccentric spheres models of the head. *IEEE Trans. Biomed. Eng., 38 No 9, pp. 871-878,1991*

Demas J.N. and Demas S.E. Interfacing and scientific computing on personal computers. Allyn and Bacon, Publishers. 1990

Einola H., Nenonen J. and Katila T., Localization studies with multichannel magnetometers. 20th International Congress on Electrocardiology, Slovak Republic May 1993.

Erné S., Trahms L. and Trontelj Z., Current multipole s as sources of biomagnetic fields. Biomagnetism '87, 6th International Conference on Biomagnetism, Tokyo, Japan Aug 1987

Ferguson A.S. and Stroink G., Localization of epicardial sources using magnetic and potential maps. In *Biomagnetism: Fundamental research and clinical applications, Baumgartner, C. et al (Eds) Elsevier Science, IOS Press 1995.*

Fiesler, T., Ioannides, A.A., Liu, M.J. and Nowak, H. Model studies of the accuracy of the conducting sphere model in magnetoencephalography using the spheroid. In *Biomagnetism: Fundamental research and clinical applications, Baumgartner, C. et al (Eds) Elsevier Science, IOS Press 1995.*

Fiesler T., An analytical expression for the magnetic field generated by a current quadrupole inside a homogeneous conducting sphere. In *'Quantitative and Topological EEG and MEG Analysis. E. Eiselt, U. Zwiener and H. Witte, eds. Universitätsverlag Druckhaus-Meyer GmbH, Jena, pp103-106,1995*

Gabor, D. And Nelson, C.V., Determination of the resultant dipole of the heart from measurements on the body surface, *J. Appl. Phys. 25 pp 413-416 1954*

Geselowitz, D.B. and Ishiwatari, H. A theoretic study of the effect of intracavitary blood mass on the dipolarity of an equivalent heart generator, In *Vectorcardiography - 1965, Hoffman,I. and Taylor, R. Eds., North-Holland, Amsterdam, 1966,393.*

Geselowitz, D.B. and Miller, W.T. Extracorporeal magnetic fields generated by internal bioelectric sources *IEEE Trans. Mag. 9, 392-398, 1973*

Geselowitz D.B., Multipole representation for an equivalent cardiac generator in volume conductors. *Proc. IRE 48, pp. 75-79, 1960*

Geselowitz, D.B. ,Two theorems concerning the quadrupole applicable to ECG, *IEEE Trans. Biomed. Eng., 12, 164, 1965*

Geselowitz D.B., On bioelectric potentials in an inhomogeneous volume conductor. *Biophys. J., 7, pp. 1-11,1967*

Geselowitz D.B., On the magnetic field generated outside an inhomogeneous volume conductor by internal current sources. *IEEE Trans. Mag., 6 No 2, pp. 346-347, 1970*

Geselowitz D.B., Magnetocardiography – An overview. *IEEE Trans. Biomed. Eng., 26 No 9, pp. 497-504,1979*

Geselowitz D.B., Computer simulation of the human magnetocardiogram. *IEEE Trans. Mag., v 16 No \5, pp. 812-817, 1980*

Golub G.H. and Pereyra V., The differentiation of pseudo-inverses and non-linear least squares problems whose variables separate. *SIAM J. Numer. Anal., 10 No2, pp. 413-432, 1973*

Gomes J., Mehra R., Barreca P., El-Sherif N., Hariman R. and Holtzman M.S., Quantitative analysis of the signal-avergaed QRS complex in patients with acute myocardial infarction: a prospective study. *Circulation, v 72 No 1, pp. 105-110, 1985*

Gonelli R.S., The "best" localization of biomagneteic sources. In *Biomagnetic Localization and 3D Modelling, Nenonen J., Rajala H. and Katila T (Eds) , Dept . of Technical Physics, Helsinki University of Technology, Finland, 1991*

Gonelli R.S. and Agnello M., Inverse problem solution in cardiomagnetism using a current multipole expansion of the primary source. *Phys. Med. Biol., v 32 No 1, pp. 133-142, 1987*

Grynszpan F. , 1971. Relationship between the surface electromagnetic fields and the electrical activity of the heart. *PhD dissertation, University of Pennsylvania, Philadelphia.*

Grynszpan, F. and Geselowitz, D.B. Model studies of the magnetocardiogram *Biophys. J., 13, 911, 1973.*

Gulrajani R.M., The forward problem of electrocardiography: from heart models to body surface potentials. *IEEE Proceedings of the 19th International Conference EMBS Chicago, IL. USA, Oct1997.*

Hasson R., Mixed EEG/MEG imaging: a way forward. In *Biomagnetic Localization and 3D Modelling, Nenonen J., Rajala H. and Katila T (Eds) , Dept . of Technical Physics, Helsinki University of Technology, Finland, 1991*

Horacek B.M., Purcell C.J., Lamothe R., Leon J., Merritt R., Kafer C., Dey S., Periyalwar S. and Stroink G., The effect of torso geometry on magnetocardiographic isofield maps. *Phys. Med. Biol., v35, pp. 121-124, 1987*

Huiskamp G. and van Oosterom A., The depolarization sequence of the human heart surface computed from measured body surface potentials, *IEEE Trans. Biomed. Eng., 35 No 12, pp. 1047-1058,1988*

Karp P.J., Katila T.E., Saarinen M., Siltanen P. and Varpula T.T., The normal human magnetocardiogram – II A multipole analysis. *Circulation Resaearch, v 47 No1 , pp.117-130, 1980*

Katila T. and Karp P., Magnetocardiography: morphology and multipole presentations: In *Biomagnetism: An Interdisciplinary Approach, Williamson S.J., Romani G., Kaufman l. and Modena I., (Eds), Plenum Press, New York and London, pp. 237-263, 1983*

Kaufman L., A variable projection method for solving separable non-linear least squares problems. *BIT, 15, pp. 49-57, 1975*

Kavenagh R.N., Darcey T.M., Lehman D. and Fender D.H., Evaluation of methods for three-dimensional localization of electrical sources in the brain. *IEEE Trans. Biomed. Eng., 25, pp. 421-429,1978*

Killmann R., Weismuller P. and Achenbach S., The relationship between size of cardiac excitation wavefront, magnetic field and dipole strength in MCG measurements, In

Biomagnetism: Fundamental research and clinical applications, Baumgartner, C. et al (Eds) Elsevier Science, IOS Press 1995.

Lin S., Roth B.J. and Wikswo J.P. Jr, Quatrefoil reentry in myocardium: an optical imaging study of the induction mechanism. *J. Cardiovasc. Electrophysiol. ,v 10, pp. 574-586, 1999*

Lütkenhöner B., Current dipole localization with an ideal magnetometer system, *IEEE Trans. Biomed. Eng., 43 No 11, pp. 1049-1061, 1996*

Mäkijärvi M., Montonen J., Toivonen L., Siltanen P., Nieminen M., Leiniö M. and Katila T., Identification of patients with myocardial infarction by high-resolution magnetocardiography and electrocardiography. *J. Electrocardiography, v 26 No 2, pp. 117-124, 1993*

Malmivuo J., Suihko V. and Eskola H., Sensitivity distributions of EEG and MEG measurements, *IEEE Trans. Biomed. Eng., 44 No 3, pp. 196-208, 1997*

Marquardt,D.W., An algorithm for least-squares estimation of nonlinear parameters, *J. Soc. Indust. Appl. Math., 11, 2, 1963*

McFee R. and Baule G., Research in electrocardiography and magnetocardiography. *Proceedings of the IEEE, v 60 no 3, pp.260-321, 1972*

McLeod R.S., Gardner M., Miller R.M. and Horácek B.M., Application of an electrocardiographic inverse solution to localize ischemia during coronary angioplasty. *J. Card. Electrophys. v 6 No1 , pp. 2-18, 1995*

Morse, P.M. and Feshbach, H. "Methods of Theoretical Physics" , *McGraw-Hill, New York, 1953*

Moshage W. and Achenbach S., Functional localization in cardiology with MCG, In *Biomagnetism: Fundamental research and clinical applications, Baumgartner, C. et al (Eds) Elsevier Science, IOS Press 1995.*

Muller H-P., Gödde P., Czerski K., Oeff M., Agrawal R., Endt P., Kruse W., Steinhoff U. and Trahms L. , Magnetocardiographic analysis of the two-dimensional distribution of intra-QRS fractionated activation. *Phys. Med. Biol., v 44, pp. 105-120, 1999*

Nenonen, J. and Katila, T., Non-invasive functional localization by biomagnetic methods, *J. Clinical Eng., v 16 No 6, 495-503, 1991*

Nenonen J., Katila T., Leiniö M., Montonen J., Mäkijärvi M., and Siltanen P., Magnetocardiographic functional localization using current multpole models. *IEEE Trans. Biomed. Eng., 38 No 7, pp. 648-657,1991*

Nenonen J., Mäkijärvi M., Toivonen L., Forsman K., Leiniö M., Montonen J., Järvinene A., Keto P., Hekali P., Katila T. and Siltanen P., Non-invasive magnetocardiographic localization
of ventricular pre-excitation in the Wolff-Parkinson-White syndrome using a realistic torso model. *European Heart J., v 14, pp. 168-174, 1993*

Nenonen J., Hämäläinen M.and Ilmoniemi R., Minimum-norm estimation in a boundary element torso model. *Medical and Biological Eng. and Comp., v 32 No 1, pp. 43-48, 1994*

Nenonen, J., Montonen, J. and Katila, T., Thermal noise in biomagnetic measurements, *Rev. Sci. Instrum., 67 (6), 2397-2405, 1996*

Noble B., Methods for computing the Moore-Penrose gneralized inverse, and related matters, In *'Generalized Inverses and Applications – Proc. of an Advanced Seminar, sponsored by the Mathematics Research Center, The University of Wisconsin-Madison, USA. Oct 1973'*

Nolte G. and Curio G., On the calculation of magnetic fileds based on multipole modeling of focal biological current sources. *Biophys. J., v 73, pp. 1253-1262, 1997*

Oostendorp T. and van Oosterom A., Decoupling linear and non-linear parameters in bioelectric source estimation, In *Engineering Solutions to Current Health Problems, Szeto A.Y. Rangayan R.M. (Eds),IEEE Publishing Services, New York, USA. pp.803-804, 1993*

Oostendorp T., McLeod R. and van Oosterom A., Non-invasive determination of the activation sequence of the heart validation with invasive data. *IEEE Proceedings of the 19th International Conference EMBS Chicago, IL. USA, Oct1997.*

Paulsen K., Finite element modeling in therapeutic and diagnostic bioelectromagnetics applications, *IEEE Proceedings of the 19th International Conference EMBS Chicago, IL. USA, Oct1997.*

Plonsey R., Capability and limitations of electrocardiography and magnetocardiography. *IEEE Trans. Biomed. Eng., 19 No 3, pp. 239-244,1972*

Plonsey R., Introductory Physics and Mathematics, In *Comprehensive Electrocardiology, McFarlane and Lawrie (Eds), Pergamon Press, New York and Oxford, pp. 41-76, 1989*

Purcell C.J., A comparison of electric and magnetic body surface mapping using single moving dipole inverse solutions, *PhD dissertation, Dalhousie University, Halifax, Nova Scotia, Canada, March 1988*

Rudy Y. and Plonsey R., The eccentric spheres model as the basis for a study of the role of geometry and inhomogeneities in electrocardiography . *IEEE Trans. Biomed. Eng., 26 No 7, pp. 392-399,1979*

Sanfelici S., A computer study of electrogram fractionation, *Electrocardiology '98, Proceedings of the XXV International Congress on Electrocardiology, Budapest, Hungary, June 1998*

Saumarez R.C., Heald S., Gill J., Slade A.K.B., de Belder M., Wallczak F., Rowland E., Ward D.E. and Camm A.J., Primary ventricular fibrillation is associated with increased paced right ventricular electrogram fractionation, *Circulation, v 92 No 9, pp.2565-2571, 1995*

Saumarez R.C., Slade A.K.B., Grace A.A., Sadoul N., Camm A.J. and McKenna W.J., The significance of paced electrogram fractionation in hypertrophic cardiomyopathy, *Circulation, v 91 No 11, pp.2762-2768, 1995*

Schoels W., Yang H., Gough W. and El-Sherif N., Circus movement atrial flutter in the canine sterile pericarditis model – activation patterns during initiation, termination and sustained reentry in vivo. *Circulation Research, v 68 No 4, pp. 1117-1126, 1991*

Schoels W., Gough W., Restivo M. and El-Sherif N., Circus movement atrial flutter in the canine sterile pericarditis model – differntial effects of procainamide on the components of the reentry pathway. *Circulation Research, v 67 No 1, pp. 35-50, 1990*

Startt/Selvester R.H., Wagner G.S. and Ideker R.E., Myocardial Infarction, In *Comprehensive Electrocardiology, McFarlane and Lawrie (Eds), Pergamon Press, New York and Oxford, pp. 565-629, 1989*

Stok, C.J., The influence of model parameters on EEG/MEG single dipole source location. *IEEE Trans. Biomed. Eng., 34,289-296,1987*

Stratton, J.A. "Electromagnetic Theory" *McGraw-Hill, New York*

Titomir, L.I. And Kneppo, P., "Bioelectric and Biomagnetic Fields - Theory and Applications in Electrocardiology", *CRC Publications, London, 1994*

Trahms L., Burghoff M., Brockmeir K. and Scmitz L., Vortex current detected by the magnetocardiogram, *Biomag 96, Proceedings of the 10th International Conference on Biomagnetism, Santa Fe, USA, 1996*

van Oosterom A., Modeling the cardiac electric field. *Medical & Biological Engineering and Computing, v 34 Supplement 1 part 2, 1996, Proceedings of the 1st International Conference on Bioelectromagnetism, June 9th 1996, Tampere, Finland.*

Vetterling W.T., Teukolsky S.A., Press W.H. and Flannery B.P., "Numerical Recipes in C", Cambridge University Press, 1985

Weismuller P., Role of magnetocardiography in cardiology, In *Biomagnetism: Fundamental research and clinical applications, Baumgartner, C. et al (Eds) Elsevier Science, IOS Press 1995.*

Weiss I., Urbaszek A and Schaldach M., New aspects of arrhythmogenesis investigated in a model of ead-induced (early after-depolarization) cardiac reentry. *IEEE Proceedings of the 19th International Conference EMBS Chicago, IL. USA, Oct1997.*

Wikswo, J.P. jr and Swinney, K.R. A comparison of multipole expansions, *J. Appl. Phys. 56, pp3039-3049,1984*

Yeh, G.C.K. and Martinek, J. The potential of a general dipole in a homogeneous conducting prolate spheroid *Ann. N.Y.Acad. Sci. 65, 1003, 1957.*

Zimmermann R. and Erné S.N., Volume conductor contribution for different body geometries. In *Biomagnetic Localization and 3D Modelling, Nenonen J., Rajala H. and Katila T (Eds), Dept. of Technical Physics, Helsinki University of Technology, Finland, 1991*

The Non-invasive Localization of Cardiac Arrythmias – a Theoretical Study

Appendices 1 - 8

28/7/99

Appendix 1: The Surface Integration for the Magnetic Vector Potential on the Surface of a Sphere

The integral to be carried out is:

$$\bar{A}(\bar{r}) = \frac{\mu\sigma}{4\pi}\int_{\varphi=0}^{2\pi}\int_{\theta=0}^{\pi}\left[\frac{1}{4\pi\sigma}\sum_{n=1}^{\infty}\sum_{m=0}^{n}\frac{2n+1}{n}\frac{\lambda_{nm}}{R^{n+1}}P_n^m(\cos\theta)\begin{Bmatrix}\cos m\varphi\\ \sin m\varphi\end{Bmatrix}\right]\cdot$$
$$\left[\frac{1}{R}\sum_{n'}^{\infty}\sum_{m'=0}^{n'}\eta'\cos\left(m[\varphi-\varphi_e]\right)P_{n'}^{m'}(\cos\theta)P_{n'}^{m'}(\cos\theta_e)\right]\cdot \qquad \text{A1.1}$$
$$\left(\bar{u}_x\sin\theta\cos\varphi + \bar{u}_y\sin\theta\sin\varphi + \bar{u}_z\cos\theta\right)R^2\sin\theta\ d\theta\ d\varphi$$

In the following, the $\begin{Bmatrix}\cos m\varphi\\ \sin m\varphi\end{Bmatrix}$ odd and even terms are treated separately since they are ultimately recombined with the odd and even terms inside λ_{nm}. They are referred to respectively as the A and B terms.

A1.1 The Integration with respect to φ

- The first part of the $\bar{u}_x$ term (the A terms)

$$\int_0^{2\pi}\cos\varphi\ \cos m\varphi\ \cos m'(\varphi-\varphi_e)\ d\varphi \qquad \text{A1.2}$$

which, because $\int_0^{2\pi}\sin m'\varphi\ d\varphi = 0$ for all $m' \geq 0$, reduces to

$$\int_0^{2\pi}\cos\varphi\ \frac{1}{2}\left[\cos(m-m')\varphi + \cos(m+m')\varphi\right]\cos m'\varphi_e d\varphi \qquad \text{A1.3}$$

$$= \int_0^{2\pi}\frac{1}{2}\left\{\begin{matrix}\left[\frac{1}{2}\left[\cos(1-m+m')\varphi + \cos(1+m-m')\varphi\right] + \right.\\ \left.\left[\frac{1}{2}\left[\cos(1-m-m')\varphi + \cos(1+m+m')\varphi\right]\right.\right.\end{matrix}\right\}\cos m'\varphi_e d\varphi \qquad \text{A1.4}$$

$$= \pi\begin{Bmatrix}1\\ \cos\varphi_e\end{Bmatrix}\begin{cases}m=1\ m'=0\\ m=0\ m'=1\end{cases} \qquad \text{A1.5}$$

$$= \frac{\pi}{2} \begin{Bmatrix} \cos(m-1)\varphi_e \\ \cos(m+1)\varphi_e \end{Bmatrix} \quad \begin{cases} m = m' + 1 \\ m' = m + 1 \end{cases} \qquad \text{A1.6}$$

- The second part of the $\overline{u_x}$ term (the B terms)

$$\int_0^{2\pi} \cos\varphi \ \sin m\varphi \ \cos m'(\varphi - \varphi_e) \ d\varphi \qquad \text{A1.7}$$

which reduces to

$$\int_0^{2\pi} \cos\varphi \ \frac{1}{2}\left[\cos(m-m')\varphi \ - \ \cos(m+m')\varphi\right] \sin m'\varphi_e d\varphi \qquad \text{A1.8}$$

$$= \int_0^{2\pi} \frac{1}{2} \left\{ \begin{matrix} \left[\frac{1}{2}\left[\cos(1-m+m')\varphi + \ \cos(1+m-m')\varphi\right] - \right. \\ \left. \left[\frac{1}{2}\left[\cos(1-m-m')\varphi + \ \cos(1+m+m')\varphi\right] \right. \right. \end{matrix} \right\} \sin m'\varphi_e d\varphi \qquad \text{A1.9}$$

$$= 0 \quad \begin{cases} m = 1 \ m' = 0 \\ m = 0 \ m' = 1 \end{cases} \qquad \text{A1.10}$$

$$= \frac{\pi}{2} \begin{Bmatrix} \sin(m-1)\varphi_e \\ \sin(m+1)\varphi_e \end{Bmatrix} \quad \begin{cases} m = m' + 1 \\ m' = m + 1 \end{cases} \qquad \text{A1.11}$$

- The first part of the $\overline{u_y}$ term (the A terms)

$$\int_0^{2\pi} \sin\varphi \ \sin m\varphi \ \cos m'(\varphi - \varphi_e) \ d\varphi \qquad \text{A1.12}$$

which reduces to

$$\int_0^{2\pi} \sin\varphi \ \frac{1}{2}\left[\sin(m-m')\varphi + \ \sin(m+m')\varphi\right] \cos m'\varphi_e d\varphi \qquad \text{A1.13}$$

$$= \int_0^{2\pi} \frac{1}{2} \left\{ \begin{matrix} \left[\frac{1}{2}\left[\cos(1-m+m')\varphi - \ \cos(1+m-m')\varphi\right] + \right. \\ \left. \left[\frac{1}{2}\left[\cos(1-m-m')\varphi - \ \cos(1+m+m')\varphi\right] \right. \right. \end{matrix} \right\} \cos m'\varphi_e d\varphi \qquad \text{A1.14}$$

$$= \pi \quad \textit{for} \ \ m = 1 \ \ m' = 0$$
$$= 0 \quad \textit{for} \ \ m = 0 \ \ m' = 1 \qquad \text{A1.16}$$

$$= \frac{\pi}{2}\cos(m-1)\varphi_e \quad for\ \ m = m'+1$$
$$= -\frac{\pi}{2}\cos(m+1)\varphi_e \quad for\ \ m' = m+1 \qquad \text{A1.17}$$

- The second part of the $\overline{u_y}$ term (the B terms)

$$\int_0^{2\pi} \sin\varphi\ \cos m\varphi\ \cos m'(\varphi-\varphi_e)\ d\varphi \qquad \text{A1.18}$$

which reduces to

$$\int_0^{2\pi} \sin\varphi\ \frac{1}{2}\left[\sin(m'-m)\varphi + \sin(m'+m)\varphi\right]\ \sin m'\varphi_e d\varphi \qquad \text{A1.19}$$

$$= \int_0^{2\pi} \frac{1}{2}\left\{\begin{array}{l}\left[\frac{1}{2}\left[\cos(1-m'+m)\varphi - \cos(1+m'-m)\varphi\right] - \right. \\ \left.\left[\frac{1}{2}\left[\cos(1-m'-m)\varphi - \cos(1+m'+m)\varphi\right]\right.\right.\end{array}\right\} \sin m'\varphi_e d\varphi \qquad \text{A1.20}$$

$$= 0 \quad for\ \ m = 1\ \ m' = 0$$
$$= \pi\ \sin\varphi_e \quad for\ \ m = 0\ \ m' = 1 \qquad \text{A1.21}$$

$$= -\frac{\pi}{2}\ \sin(m-1)\varphi_e \quad for\ \ m = m'+1$$
$$= \frac{\pi}{2}\ \sin(m+1)\varphi_e \quad for\ \ m' = m+1 \qquad \text{A1.22}$$

- The first part of the $\overline{u_z}$ term (the A terms)

$$\int_0^{2\pi} \cos m\varphi\ \cos m'(\varphi-\varphi_e)\ d\varphi \qquad \text{A1.23}$$

which reduces to

$$\int_0^{2\pi} \frac{1}{2}\left[\cos(m-m')\varphi + \cos(m+m')\varphi\right]\ \cos m'\varphi_e d\varphi \qquad \text{A1.24}$$

$$= 2\pi \quad for\ \ m = m' = 0$$
$$= \pi\ \cos m\varphi_e \quad for\ \ m = m' \qquad \text{A1.25}$$

- The second part of the $\overline{u_z}$ term (the B terms)

$$\int_0^{2\pi} \sin m\varphi \cos m'(\varphi - \varphi_e)\, d\varphi \qquad \text{A1.26}$$

which reduces to

$$\int_0^{2\pi} \frac{1}{2}\left[\cos(m-m')\varphi - \cos(m+m')\varphi\right] \sin m'\varphi_e d\varphi \qquad \text{A1.27}$$

$$\begin{aligned} &= 0 && \text{for } m = m' = 0 \\ &= \pi \sin m\varphi_e && \text{for } m = m' \end{aligned} \qquad \text{A1.28}$$

A1.2 The Integration with respect to θ

The integration with the θ dependent elements is:

$$\int_0^{\pi} \left[\sum_{n=1}^{\infty}\sum_{m=0}^{n}\lambda_{nm}P_n^m(\cos\theta)\right]\left[\sum_{n'=0}^{\infty}\sum_{m'=0}^{n'}\eta_{n'}^{m'}P_{n'}^{m'}(\cos\theta)\right]\cdot \left[(\overline{u_x} + \overline{u_y})\sin\theta + \overline{u_z}\cos\theta\right]\sin\theta\ P_{n'}^{m'}(\cos\theta_e)\, d\theta \qquad \text{A1.29}$$

where the $\lambda_{nm}, \eta_{n'}^{m'}$ *and* θ_e terms are constant.

- The $(\overline{u_x} + \overline{u_y})$ terms

$$\int_0^{\pi} \left[\sum_{n=1}^{\infty}\sum_{m=0}^{n}\lambda_{nm}P_n^m(\cos\theta)\right]\left[\sum_{n'=0}^{\infty}\sum_{m'=0}^{n'}\eta_{n'}^{m'}P_{n'}^{m'}(\cos\theta)P_{n'}^{m'}(\cos\theta_e)\right]\cdot \sin\theta\ \sin\theta\ d\theta \qquad \text{A1.30}$$

From the previous section $m = m' \pm 1$ Omitting the $\cos\theta$ in the Legendre expressions and the summation signs, for the sake of brevity, and using recursion identities shown in Appendix .2

for $m = m' + 1$

$$\int_0^{\pi} \lambda_{nm}\ P_n^m\ \eta_{n'}^{m'}\ P_{n'}^{m-1}\sin\theta\ \sin\theta\ P_{n'}^{m-1}(\cos\theta_e)d\theta \qquad \text{A1.31}$$

$$= \int_0^{\pi} \lambda_{nm} \; P_n^m \; \eta_{n'}^{m'} \left[P_{n'+1}^m - P_{n'-1}^m \right] \sin\theta \; \frac{1}{2n'+1} \; P_{n'}^{m-1}(\cos\theta_e) \; d\theta \qquad \text{A1.32}$$

which becomes:

$$\sum_{n=1}^{\infty} \lambda_{nm} \frac{2}{2n+1} \left(2 - \delta_m^1\right) \left(\sum_{m=0}^{n} \frac{(n+m)(n+m-1)}{2n-1} P_{n-1}^{m-1}(\cos\theta_e) - \sum_{m=1}^{n+2} \frac{(n+m+2)(n+m+1)}{2n+3} P_{n+1}^{m-1}(\cos\theta_e) \right) \qquad \text{A1.33}$$

for $m = m' - 1$

from the identities in Appendix 2. the following relationships can be derived:

$$(n+m)(\sin\theta) P_n^{m-1} = P_{n+1}^m - (\cos\theta) P_n^m \qquad \text{A1.34}$$

$$(\sin\theta) P_n^{m+1} = (n+m+1)(\cos\theta) P_n^m - (n-m+1) P_{n+1}^m \qquad \text{A1.35}$$

the integration is:

$$\int_0^{\pi} \lambda_{nm} \; P_n^m \; \eta_{n'}^{m'} \; P_{n'}^{m+1} \sin\theta \; \sin\theta \; P_{n'}^{m+1}(\cos\theta_e) d\theta \qquad \text{A1.36}$$

making the appropriate substitutions this becomes:

$$\int_0^{\pi} \lambda_{nm} \; P_n^m \; \eta_{n'}^{m'} \frac{1}{2n'+1} \left[\begin{matrix} (n'+m)(n'+m+1) P_{n'-1}^m - \\ (n'-m)(n'-m+1) P_{n'+1}^m \end{matrix} \right] \cdot \sin\theta \; P_{n'}^{m+1}(\cos\theta_e) d\theta \qquad \text{A1.37}$$

$$= \int_0^{\pi} \lambda_{nm} \; P_n^m \left[\begin{matrix} \eta_{n+1}^{m+1} \frac{(n+m+1)(n+m+2)}{2n+3} P_n^m - \\ \eta_{n-1}^{m+1} \frac{(n-m-1)(n-m)}{2n-1} P_n^m \end{matrix} \right] \sin\theta \; P_{n'}^{m+1}(\cos\theta_e) d\theta \qquad \text{A1.38}$$

which becomes:

$$= \sum_{n=1}^{\infty} \lambda_{nm} \frac{4}{2n+1} \left(\sum_{m=0}^{n} \frac{P_{n+1}^{m+1}(\cos\theta_e)}{2n+3} - \sum_{m=0}^{n-2} \frac{P_{n-1}^{m+1}(\cos\theta_e)}{2n-1} \right) \qquad \text{A1.39}$$

- The $(\overline{u_z})$ term

$$m = m'$$

$$\int_0^{\pi} \left[\sum_{n=1}^{\infty} \sum_{m=0}^{n} \lambda_{nm} P_n^m(\cos\theta)\right] \cdot \left[\sum_{n'=0}^{\infty} \sum_{m'=0}^{n'} \eta_{n'}^{m'} P_{n'}^{m'}(\cos\theta) P_{n'}^{m'}(\cos\theta_e)\right] \cos\theta \sin\theta \, d\theta \qquad \text{A1.40}$$

$$= \int_0^{\pi} \left[\sum_{n=1}^{\infty} \sum_{m=0}^{n} \lambda_{nm} P_n^m(\cos\theta)\right] \cdot \left[\sum_{n'=0}^{\infty} \sum_{m'=0}^{n'} \frac{\eta_{n'}^{m'}}{2n'+1} \begin{Bmatrix} (n'+m') P_{n'-1}^{m'}(\cos\theta) + \\ (n'-m'+1) P_{n'+1}^{m'}(\cos\theta) \end{Bmatrix} P_{n'}^{m'}(\cos\theta_e)\right] \sin\theta \, d\theta \qquad \text{A1.41}$$

which becomes:

$$\sum_{n=1}^{\infty} \lambda_{nm} \frac{2}{2n+1}\left(2-\delta_m^0\right) \begin{pmatrix} \sum_{m=0}^{n+1} \frac{(n-m+1)}{2n+3} P_{n+1}^m(\cos\theta_e) + \\ \sum_{m=0}^{n-1} \frac{(n+m)}{2n-1} P_{n-1}^m(\cos\theta_e) \end{pmatrix} \qquad \text{A1.42}$$

A1.3 The Two Integrations are Combined

Combining the two results together with the inner $\frac{1}{4\pi}$ gives (using f_α to denote the series relating to the α component of the vector):

- The $(\overline{u_x} + \overline{u_y})$ terms

the $m = m' + 1$ series

$$\sum_{n=1}^{\infty} \left\| \begin{matrix} f_x \langle A \rangle \\ f_y \langle B \rangle \end{matrix} \right\|_{m'+1} = \sum_{n=1}^{\infty} \frac{1}{2(2n+1)} \cdot \begin{pmatrix} \sum_{m=0}^{n} \lambda_{nm} \frac{(n+m)(n+m-1)}{2n-1} P_{n-1}^{m-1}(\cos\theta_e) - \\ \sum_{m=1}^{n+2} \lambda_{nm} \frac{(n+m+2)(n+m+1)}{2n+3} P_{n+1}^{m-1}(\cos\theta_e) \end{pmatrix} \left\langle \begin{matrix} \cos(m-1)\varphi_e \\ \pm \sin(m-1)\varphi_e \end{matrix} \right\rangle \qquad \text{A1.43}$$

the $m = m' - 1$ series

$$\sum_{n=1}^{\infty} f_x \left\langle \begin{matrix} A \\ B \end{matrix} \right\rangle_{m'-1} = \sum_{n=1}^{\infty} \frac{1}{(2n+1)} \cdot \begin{pmatrix} \sum_{m=0}^{n} \frac{\lambda_{nm}}{\left(2-\delta_m^0\right)} \frac{P_{n+1}^{m+1}(\cos\theta_e)}{2n+3} \\ \sum_{m=0}^{n-2} \frac{\lambda_{nm}}{\left(2-\delta_m^0\right)} \frac{P_{n-1}^{m+1}(\cos\theta_e)}{2n-1} \end{pmatrix} \left\langle \begin{matrix} \cos(m+1)\varphi_e \\ \left(1-\delta_m^0\right)\sin(m+1)\varphi_e \end{matrix} \right\rangle \quad \text{A1.44}$$

$$\sum_{n=1}^{\infty} f_y \left\langle \begin{matrix} A \\ B \end{matrix} \right\rangle_{m'-1} = \sum_{n=1}^{\infty} \frac{1}{(2n+1)} \cdot \begin{pmatrix} \sum_{m=0}^{n} \frac{\lambda_{nm}}{\left(2-\delta_m^0\right)} \frac{P_{n+1}^{m+1}(\cos\theta_e)}{2n+3} \\ \sum_{m=0}^{n-2} \frac{\lambda_{nm}}{\left(2-\delta_m^0\right)} \frac{P_{n-1}^{m+1}(\cos\theta_e)}{2n-1} \end{pmatrix} \left\langle \begin{matrix} -\left(1-\delta_m^0\right)\cos(m+1)\varphi_e \\ \sin(m+1)\varphi_e \end{matrix} \right\rangle \quad \text{A1.45}$$

- The $\left(\overline{u_z}\right)$ term

the $m = m'$ series

$$\sum_{n=1}^{\infty} f_z \left\langle \begin{matrix} A \\ B \end{matrix} \right\rangle = \sum_{n=1}^{\infty} \sum_{m=0}^{n} \lambda_{nm} \frac{1}{(2n+1)} \cdot \begin{pmatrix} \frac{(n-m+1)}{2n+3} P_{n+1}^{m}(\cos\theta_e) + \\ \frac{(n+m)}{2n-1} P_{n-1}^{m}(\cos\theta_e) \end{pmatrix} \left\langle \begin{matrix} \cos m\varphi_e \\ \sin m\varphi_e \end{matrix} \right\rangle \quad \text{A1.46}$$

Appendix 2. Associated Legendre Relationships

The following notation is used $\frac{d}{d\theta}P_n^m(\cos\theta) = P_n'^m(\cos\theta)$

$$\int_0^\pi P_n^m(\cos\theta)P_{n'}^m(\cos\theta)\sin\theta\, d\theta \begin{cases} = \frac{2}{2n+1}\frac{(n+m)!}{(n-m)!} & for\ n' = n \\ = 0 & for\ n' \neq n \end{cases} \quad \text{A2.1}$$

$$\frac{d}{d(\cos\theta)}P_n^m(\cos\theta) = -\frac{1}{\sin\theta}P_n'^m(\cos\theta) \quad \text{A2.2}$$

$$P_n'^m(\cos\theta) = m\frac{\cos\theta}{\sin\theta}P_n^m(\cos\theta) - P_n^{m+1}(\cos\theta) \quad \text{A2.3}$$

$$P_n'^m(\cos\theta) = \frac{(n+m)(n-m+1)}{2}P_n^{m-1}(\cos\theta) - \frac{1}{2}P_n^{m+1}(\cos\theta) \quad \text{A2.4}$$

$$P_{n+1}'^m(\cos\theta) = \frac{(n-m+1)}{\sin\theta}P_n^m(\cos\theta) - (n+1)\frac{\cos\theta}{\sin\theta}P_{n+1}^m(\cos\theta) \quad \text{A2.5}$$

$$P_{n-1}'^m(\cos\theta) = \frac{(n-m)}{\sin\theta}P_n^m(\cos\theta) - n\frac{\cos\theta}{\sin\theta}P_{n-1}^m(\cos\theta) \quad \text{A2.6}$$

$$2m\frac{\cos\theta}{\sin\theta}P_n^m(\cos\theta) = P_n^{m+1}(\cos\theta) + (n+m)(n-m+1)P_n^{m-1}(\cos\theta) \quad \text{A2.7}$$

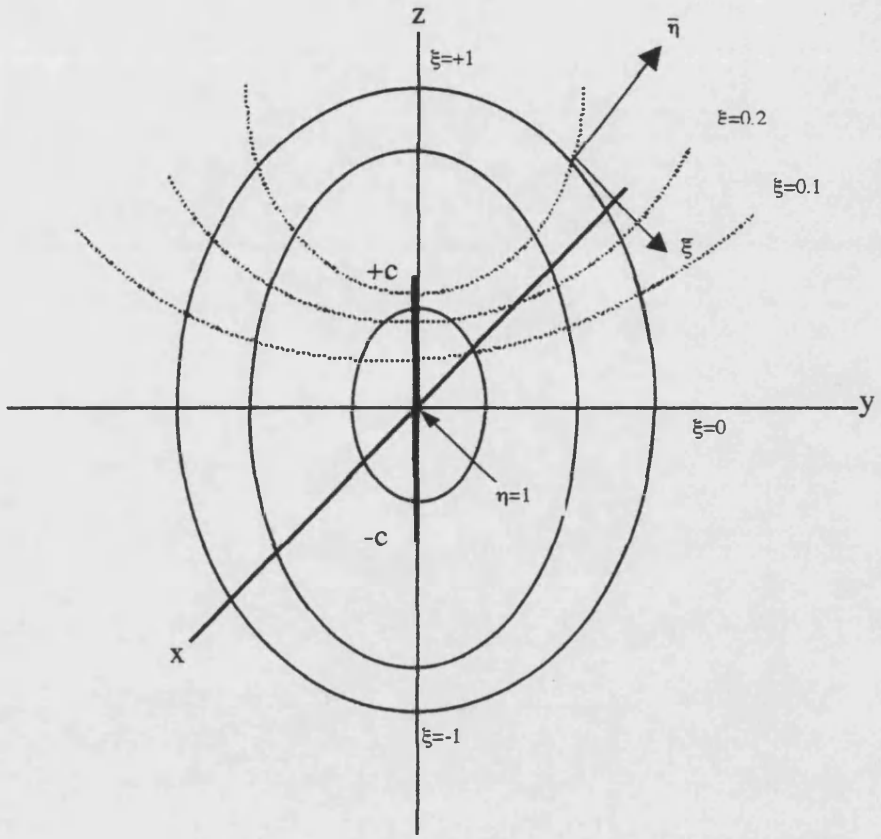
$$(2n+1)\cos\theta\ P_n^m(\cos\theta) = (n+m)P_{n-1}^m(\cos\theta) + (n-m+1)P_{n+1}^m(\cos\theta) \quad \text{A2.8}$$

$$(2n+1)\sin\theta\ P_n^m(\cos\theta) = P_{n+1}^{m+1}(\cos\theta) - P_{n-1}^{m+1}(\cos\theta) \quad \text{A2.9}$$

$$P_n^{-m}(\cos\theta) = (-1)^m\frac{(n-m)!}{(n+m)!}P_n^m(\cos\theta) \quad \text{A2.10}$$

Appendix 3. Details of the Prolate Spheroid Equations

Appendix 3.1 – Prolate Spheroid Co-ordinate System



The right handed co-ordinate system (η,ξ,φ) defines the prolate spheroid which consists of confocal ellipses of constant η, hyperbolas of constant ξ rotated from x to y about z and the azimuth angle φ measured from x to y. The co-ordinate ranges are:

$$1 \le \eta \le \infty \quad -1 \le \xi \le +1 \quad 0 \le \varphi \le 2\pi$$

The metric coefficients are:

$$h_\eta = c\frac{\left(\eta^2-\xi^2\right)^{1/2}}{\left(\eta^2-1\right)^{1/2}} \quad h_\xi = c\frac{\left(\eta^2-\xi^2\right)^{1/2}}{\left(1-\xi^2\right)^{1/2}} \quad h_\varphi = c\left(\eta^2-1\right)^{1/2}\left(1-\xi^2\right)^{1/2} \qquad \text{A3.1}$$

The equations describing the prolate spheroid are the spheroid equation:

$$\frac{\rho^2}{c^2\left(\eta^2-1\right)} + \frac{z^2}{c^2\eta^2} = 1 \qquad \text{A3.2}$$

and the hyperboloid equation:

$$-\frac{\rho^2}{c^2\left(1-\xi^2\right)} + \frac{z^2}{c^2\xi^2} = 1 \qquad \text{A3.3}$$

where ρ, the cylindrical radius on the xy plane $= c\left(\eta^2-1\right)^{1/2}\left(1-\xi^2\right)^{1/2}$ A3.4

Any radius r from the origin $r = c\left(\eta^2+\xi^2-1\right)^{1/2}$ A3.5

The rectangular co-ordinates are:

$$\begin{Bmatrix} x \\ y \end{Bmatrix} = c\left(\eta^2-1\right)^{1/2}\left(1-\xi^2\right)^{1/2}\begin{Bmatrix} \cos\varphi \\ \sin\varphi \end{Bmatrix} \qquad z = c\eta\xi \qquad \text{A3.6}$$

The rectangular unit vectors may be derived from the above equations using, as an example:

$$\hat{i} = \frac{1}{h_\eta}\frac{dx}{d\eta}\hat{\eta} + \frac{1}{h_\xi}\frac{dx}{d\xi}\hat{\xi} + \frac{1}{h_\varphi}\frac{dx}{d\varphi}\hat{\varphi} \qquad \text{A3.7}$$

which gives:

$$\begin{Bmatrix} \hat{i} \\ \hat{j} \end{Bmatrix} = \left(\frac{c\eta}{h_\xi}\hat{\eta} - \frac{c\xi}{h_\eta}\hat{\xi}\right)\begin{Bmatrix} \cos\varphi \\ \sin\varphi \end{Bmatrix} + \begin{Bmatrix} -\sin\varphi \\ \cos\varphi \end{Bmatrix}\hat{\varphi} \qquad \hat{k} = \frac{c\eta}{h_\xi}\hat{\xi} + \frac{c\xi}{h_\eta}\hat{\eta} \qquad \text{A3.8}$$

The spheroidal unit vectors may be derived from the above using, as an example:

$$\hat{\eta} = \frac{\partial\bar{r}/\partial\eta}{\left|\partial\bar{r}/\partial\eta\right|} \quad \text{where } \bar{r} = c\left(\eta^2-1\right)^{1/2}\left(1-\xi^2\right)^{1/2}\begin{Bmatrix} \cos\varphi\ \hat{i} \\ \sin\varphi\ \hat{j} \end{Bmatrix} + c\eta\xi\hat{k} \qquad \text{A3.9}$$

which gives:

$$\hat{\eta} = \frac{c\eta}{h_\xi}\begin{Bmatrix} \cos\varphi\ \hat{i} \\ \sin\varphi\ \hat{j} \end{Bmatrix} + \frac{c\xi}{h_\eta}\hat{k} \qquad \hat{\xi} = -\frac{c\xi}{h_\eta}\hat{\xi}\begin{Bmatrix} \cos\varphi\ \hat{i} \\ \sin\varphi\ \hat{j} \end{Bmatrix} + \frac{c\eta}{h_\xi}\hat{k}$$

$$\hat{\varphi} = -\sin\varphi\ \hat{i} + \cos\varphi\ \hat{j} \qquad \text{A3.10}$$

A vector element of surface $\overline{ds} = \hat{\eta}\ ds = \hat{\eta} h_\varphi h_\xi\ d\varphi\ d\xi$ A3.11

and, for example, $\nabla\left(\frac{1}{R}\right) \times \overline{ds}$ =

$$\begin{bmatrix} \hat{\eta} & \hat{\xi} & \hat{\varphi} \\ \frac{1}{h_\eta}\frac{\partial}{\partial \eta}\left(\frac{1}{R}\right) & \frac{1}{h_\xi}\frac{\partial}{\partial \xi}\left(\frac{1}{R}\right) & \frac{1}{h_\varphi}\frac{\partial}{\partial \varphi}\left(\frac{1}{R}\right) \\ \left(h_\eta h_\xi \; d\varphi d\xi\right) & - & - \end{bmatrix} \qquad \text{A3.12}$$

$$= \left(\frac{1}{h_\xi}\frac{\partial}{\partial \xi}\left(\frac{1}{R}\right)\hat{\varphi} + \frac{1}{h_\varphi}\frac{\partial}{\partial \varphi}\left(\frac{1}{R}\right)\hat{\xi}\right) h_\eta h_\xi \; d\varphi d\xi \qquad \text{A3.13}$$

The gradient operator, ∇, is:

$$\frac{1}{h_\eta}\frac{\partial}{\partial \eta}\hat{\eta} + \frac{1}{h_\xi}\frac{\partial}{\partial \xi}\hat{\xi} + \frac{1}{h_\varphi}\frac{\partial}{\partial \varphi}\hat{\varphi} \qquad \text{A3.14}$$

and can be expressed in cartesian co-ordinates by the use of:

$$\begin{aligned} &\left[\frac{1}{h_\eta}\frac{c\eta}{h_\xi}\cos\varphi\frac{\partial}{\partial \eta} - \frac{1}{h_\xi}\frac{c\xi}{h_\eta}\cos\varphi\frac{\partial}{\partial \xi} - \frac{1}{h_\varphi}\sin\varphi\frac{\partial}{\partial \varphi}\right]\hat{i} \\ &\left[\frac{1}{h_\eta}\frac{c\eta}{h_\xi}\sin\varphi\frac{\partial}{\partial \eta} - \frac{1}{h_\xi}\frac{c\xi}{h_\eta}\sin\varphi\frac{\partial}{\partial \xi} + \frac{1}{h_\varphi}\cos\varphi\frac{\partial}{\partial \varphi}\right]\hat{j} \\ &\left[\frac{1}{h_\eta}\frac{c\xi}{h_\eta}\frac{\partial}{\partial \eta} - \frac{1}{h_\xi}\frac{c\eta}{h_\xi}\frac{\partial}{\partial \xi}\right]\hat{k} \end{aligned} \qquad \text{A3.15}$$

Appendix 3.2 – Multipole Coefficients for Prolate Spheroids

The infinite medium potential due to a dipole projected as a spheroidal wave onto the surface η_e of a prolate spheroid is:

$$\Phi_{(\eta,\xi)} = \frac{1}{4\pi\sigma}\sum_{n=0}^{\infty}\sum_{m=0}^{n}\left(A_{nm}\cos m\varphi + B_{nm}\sin m\varphi\right)P_n^m(\eta_e)P_n^m(\xi) \qquad \text{A3.16}$$

where the source at (η_0,ξ_0,φ_0) is described in terms of multipole coefficients:

$$\begin{Bmatrix}A_{nm}\\B_{nm}\end{Bmatrix} = \frac{1}{c}\left(2-\delta_m^0\right)(-1)^m(2n+1)\left[\frac{(n-m)!}{(n+m)!}\right]^2 \cdot \overline{D}\cdot\nabla\left[P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right] \qquad \text{A3.17}$$

The gradient of the source spatial configuration is expressed as:

$$\nabla\left[P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right] = \frac{1}{h_\eta}\left[P_n'^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right]\hat{\eta} + \frac{1}{h_\xi}\left[P_n^m(\eta_0)P_n'^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right]\hat{\xi} + \frac{1}{h_\varphi}\left[P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}-m\sin m\varphi_0\\ m\cos m\varphi_0\end{Bmatrix}\right]\hat{\varphi} \qquad \text{A3.18}$$

Therefore the $\hat{i}$ component is:

$$\frac{c\eta_0}{h_\xi}\frac{\cos\varphi_0}{h_\eta}\left[P_n'^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right] - \frac{c\xi_0}{h_\eta}\frac{\cos\varphi_0}{h_\xi}\left[P_n^m(\eta_0)P_n'^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right] - \frac{\sin\varphi_0}{h_\varphi}\left[P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}-m\sin m\varphi_0\\ m\cos m\varphi_0\end{Bmatrix}\right] \qquad \text{A3.19}$$

This is rationalised using the following identity:

$$P_n'^m(z) = \frac{(n+1)z}{1-z^2}P_n^m(z) - \frac{(n-m+1)}{1-z^2}P_{n+1}^m(z) \qquad \text{A3.20}$$

where the prime indicates the differential.

Applying this identity to:

$$\frac{c}{h_\xi}\frac{\cos\varphi_0}{h_\eta}\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\left[\eta_0 P_n'^m(\eta_0)P_n^m(\xi_0)-\xi_0 P_n^m(\eta_0)P_n'^m(\xi_0)\right] \qquad \text{A3.21}$$
$$-\frac{\sin\varphi_0}{h_\varphi}\begin{Bmatrix}-m\sin m\varphi_0\\ m\cos m\varphi_0\end{Bmatrix}P_n^m(\eta_0)P_n^m(\xi_0)$$

gives for the expression in square brackets:

$$\left\{\frac{\eta_0^2}{\left(1-\eta_0^2\right)}-\frac{\xi_0^2}{\left(1-\xi_0^2\right)}\right\}(n+1)P_n^m(\eta_0)P_n^m(\xi_0) \qquad \text{A3.22}$$
$$=-\frac{h_\eta h_\xi}{ch_\varphi}(n+1)P_n^m(\eta_0)P_n^m(\xi_0)$$

and the second part of the identity:

$$\left\{\frac{\xi_0^2}{\left(1-\xi_0^2\right)}P_n^m(\eta_0)P_{n+1}^m(\xi_0)-\frac{\eta_0^2}{\left(1-\eta_0^2\right)}P_{n+1}^m(\eta_0)P_n^m(\xi_0)\right\}(n-m+1) \qquad \text{A3.23}$$

resulting in:

$$\left[-\frac{(n+1)\cos\varphi_0}{h_\varphi}\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}+\frac{\sin\varphi_0}{h_\varphi}\begin{Bmatrix}-m\sin m\varphi_0\\ m\cos m\varphi_0\end{Bmatrix}\right]P_n^m(\eta_0)P_n^m(\xi_0)+$$
$$\left\{\frac{\xi_0^2}{\left(1-\xi_0^2\right)}P_n^m(\eta_0)P_{n+1}^m(\xi_0)-\frac{\eta_0^2}{\left(1-\eta_0^2\right)}P_{n+1}^m(\eta_0)P_n^m(\xi_0)\right\}(n-m+1)\frac{c}{h_\xi}\frac{\cos\varphi_0}{h_\varphi}\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix} \qquad \text{A3.24}$$

- The $\hat{j}$ component is the same but with $\cos\varphi_o$ replace by $\sin\varphi_o$ and with $\sin\varphi_o$ replaced by $-\cos\varphi_o$
- The $\hat{k}$ component is:

$$\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\left[\frac{c\xi_0}{h_\eta h_\eta}\frac{\eta_0}{\left(1-\eta_0^2\right)}+\frac{c\eta_0}{h_\xi h_\xi}\frac{\xi_0}{\left(1-\xi_0^2\right)}\right](n+1)P_n^m(\eta_0)P_n^m(\xi_0) \qquad \text{A3.25}$$

Applying the first part of the identity for $P'(z)$:

$$\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\left[\frac{c\xi_0}{h_\eta h_\eta}P_n'^m(\eta_0)P_n^m(\xi_0)+\frac{c\eta_0}{h_\xi h_\xi}P_n^m(\eta_0)P_n'^m(\xi_0)\right] \qquad \text{A3.26}$$

$$= \begin{Bmatrix}\cos m\varphi_0 \\ \sin m\varphi_0\end{Bmatrix}\left[-\frac{h_\eta^2}{c^2\left(\eta^2-\xi^2\right)h_\eta^2}+\frac{h_\xi^2}{c^2\left(\eta^2-\xi^2\right)h_\xi^2}\right]c\xi_0\eta_0(n+1)P_n^m(\eta_0)P_n'^m(\xi_0) \qquad \text{A3.27}$$

$$= 0$$

and the second part:

$$c\begin{Bmatrix}\cos m\varphi_0 \\ \sin m\varphi_0\end{Bmatrix}\left\{\frac{1}{\left(1-\eta_0^2\right)}\frac{\xi_0}{h_\eta^2}P_{n+1}^m(\eta_0)P_n^m(\xi_0)-\frac{1}{\left(1-\xi_0^2\right)}\frac{\eta_0}{h_\xi^2}P_n^m(\eta_0)P_{n+1}^m(\xi_0)\right\}(n-m+1)$$

$$= \frac{1}{c\left(\eta_0^2-\xi_0^2\right)}\begin{Bmatrix}\cos m\varphi_0 \\ \sin m\varphi_0\end{Bmatrix}\left\{\xi_0P_{n+1}^m(\eta_0)P_n^m(\xi_0)-\eta_0P_n^m(\eta_0)P_{n+1}^m(\xi_0)\right\}(n-m+1) \qquad \text{A3.28}$$

Component Evaluation

A_{10} A3.29

The $\hat{i}$ and $\hat{j}$ components $= 0$

The $\hat{k}$ component $= \frac{2}{c\left(\eta^2-\xi^2\right)}\left\{\xi^2\frac{\left(3\eta^2-1\right)}{2}-\eta^2\frac{\left(3\xi^2-1\right)}{2}\right\} = \frac{1}{c}$ A3.30

Thus $A_{10} = \frac{3D_z}{c^2}$ A3.31

A_{11} A3.32

The $\hat{j}$ and $\hat{k}$ components $= 0$

The $\hat{i}$ component $= \left\{-\frac{2\cos^2\varphi_0}{h_\varphi}+\frac{\sin^2\varphi_0}{h_\varphi}\right\}P_1^1(\eta_0)P_1^1(\xi_0)$

$$+\frac{c\cos^2\varphi_0}{h_\eta h_\xi}\left[\frac{3\xi_0^2}{\left(1-\xi_0^2\right)}-\frac{3\eta_0^2}{\left(1-\eta_0^2\right)}\right]P_1^1(\eta_0)P_1^1(\xi_0) \qquad \text{A3.33}$$

$$= \left\{-\frac{2\cos^2\varphi_0}{h_\varphi}+\frac{\sin^2\varphi_0}{h_\varphi}\right\}P_1^1(\eta_0)P_1^1(\xi_0)$$

$$+\frac{c\cos^2\varphi_0}{h_\eta h_\xi}\left[\frac{3\left(\xi_0^2-\eta_0^2\right)}{\left(1-\xi_0^2\right)\left(1-\eta_0^2\right)}\right]P_1^1(\eta_0)P_1^1(\xi_0)$$

$$= \left\{-\frac{2\cos^2\varphi_0}{h_\varphi} + \frac{\sin^2\varphi_0}{h_\varphi}\right\} P_1^1(\eta_0)P_1^1(\xi_0)$$

$$+ \frac{3c\cos^2\varphi_0}{h_\varphi} P_1^1(\eta_0)P_1^1(\xi_0)$$

$$= \frac{P_1^1(\eta_0)P_1^1(\xi_0)}{h_\varphi} = \frac{j}{c} \qquad \text{A3.33(cont)}$$

Thus $A_{11} = -\frac{3jD_x}{2c^2}$ A3.34

B_{11} A3.35

The $\hat{i}$ and $\hat{k}$ components $=0$

The $\hat{j}$ component $= \left\{-\frac{2\sin^2\varphi_0}{h_\varphi} + \frac{\cos^2\varphi_0}{h_\varphi}\right\} P_1^1(\eta_0)P_1^1(\xi_0)$

$$+ \frac{c\sin^2\varphi_0}{h_\eta h_\xi}\left[\frac{3\xi_0^2}{\left(1-\xi_0^2\right)} - \frac{3\eta_0^2}{\left(1-\eta_0^2\right)}\right] P_1^1(\eta_0)P_1^1(\xi_0) \qquad \text{A3.36}$$

$= \frac{j}{c}$

Thus $B_{11} = -\frac{3jD_y}{2c^2}$ A3.37

A_{20}

The $\hat{i}$ component $= \frac{c\cos\varphi_0}{h_\eta h_\xi}\left\{\eta_0 P_n'^m(\eta_0)P_n^m(\xi_0) - \xi_0 P_n^m(\eta_0)P_n'^m(\xi_0)\right\}$

$$= \frac{c\cos\varphi_0}{h_\eta h_\xi}\left[3\eta_0^2\frac{\left(3\xi_0^2-1\right)}{2} - 3\xi_0^2\frac{\left(3\eta_0^2-1\right)}{2}\right]$$

$$= -\frac{3}{2}\frac{c\cos\varphi_0}{h_\eta h_\xi}\left(\eta_0^2-\xi_0^2\right) = -\frac{3x_0}{2c} \qquad \text{A3.38}$$

The $\hat{j}$ component $= -\frac{3y_0}{2c}$ A3.39

The $\hat{k}$ component $= c\left[\frac{\xi_0}{h_\eta^2}3\eta_0\frac{\left(3\xi_0^2-1\right)}{2}-\frac{\eta_0}{h_\xi^2}3\xi_0\frac{\left(3\eta_0^2-1\right)}{2}\right]$

$$= \frac{3\eta_0\xi_0}{2c}\frac{1}{\left(\eta_0^2-\xi_0^2\right)}\left[\left(\eta_0^2-1\right)\left(3\xi_0^2-1\right)+\left(1-\xi_0^2\right)\left(3\eta_0^2-1\right)\right]$$

$$= \frac{3z_0}{c} \qquad \text{A3.40}$$

Thus $A_{20} = \frac{15}{c^2}\left(z_0 D_z - \frac{x_0 D_x + y_0 D_y}{2}\right)$ A3.41

A_{21}

The $\hat{i}$ component $= \left\{-3\cos^2\varphi_0+\sin^2\varphi_0\right\}\frac{P_2^1(\eta_0)P_2^1(\xi_0)}{h_\varphi}+$

$$\frac{2c\cos^2\varphi_0 P_2^1(\eta_0)P_2^1(\xi_0)}{h_\eta h_\xi}\left[\frac{\xi_0}{\left(1-\xi_0^2\right)}\frac{\left(5\xi_0^2-1\right)}{2\xi_0}-\frac{\eta_0}{\left(1-\eta_0^2\right)}3\xi_0\frac{\left(5\eta_0^2-1\right)}{2\eta_0}\right]$$

$$= \left\{-3\cos^2\varphi_0+\sin^2\varphi_0\right\}\frac{P_2^1(\eta_0)P_2^1(\xi_0)}{h_\varphi}+$$

$$\frac{c\cos^2\varphi_0 P_2^1(\eta_0)P_2^1(\xi_0)}{h_\eta h_\xi}\left[\frac{\left(1-\eta_0^2\right)\left(5\xi_0^2-1\right)-\left(1-\xi_0^2\right)\left(5\eta_0^2-1\right)}{\left(1-\xi_0^2\right)\left(1-\eta_0^2\right)}\right]$$

$$= \left\{-3\cos^2\varphi_0+\sin^2\varphi_0\right\}\frac{P_2^1(\eta_0)P_2^1(\xi_0)}{h_\varphi}-4\left(\xi_0^2-\eta_0^2\right)\frac{\cos^2\varphi_0 P_2^1(\eta_0)P_2^1(\xi_0)}{h_\varphi\left(\eta_0^2-\xi_0^2\right)}$$

$$= \frac{P_2^1(\eta_0)P_2^1(\xi_0)}{h_\varphi} = 9(\eta_0\xi_0)\frac{\left[\left(1-\eta_0^2\right)\left(1-\xi_0^2\right)\right]^{1/2}}{h_\varphi}$$

$$= \frac{9z_0}{c}j \qquad \text{A3.42}$$

The $\hat{j}$ component $= 0$ A3.43

The $\hat{k}$ component

$$\frac{2\cos\varphi_0}{c\left(\eta_0^2-\xi_0^2\right)}\left[\xi_0^2\frac{9\left(5\eta_0^2-1\right)\left[\left(1-\xi_0^2\right)\left(1-\eta_0^2\right)\right]^{1/2}}{2}-\eta_0^2\frac{9\left(5\xi_0^2-1\right)\left[\left(1-\xi_0^2\right)\left(1-\eta_0^2\right)\right]^{1/2}}{2}\right] \qquad \text{A3.44}$$

$$= \frac{9c\cos\ \varphi_0}{h_\eta h_\xi}\left(\eta_0^2 - \xi_0^2\right)j \ = \ \frac{9x_0}{c}j \qquad \text{A3.44(cont)}$$

$$\text{Thus } A_{21} \ = \ -\frac{5}{2c^2}\left(z_0 D_x + x_0 D_z\right) \qquad \text{A3.45}$$

A_{22}

$$\text{The } \hat{i} \text{ component } = \left\{-3\cos\varphi_0\cos 2\varphi_0 + 2\sin\varphi_0\sin 2\varphi_0\right\}\frac{P_2^2(\eta_0)P_2^2(\xi_0)}{h_\varphi} +$$

$$\frac{c\cos\varphi_0\cos 2\varphi_0 P_2^2(\eta_0)P_2^2(\xi_0)}{h_\eta h_\xi}\left[\frac{5\xi_0^2}{\left(1-\xi_0^2\right)} - \frac{5\eta_0^2}{\left(1-\eta_0^2\right)}\right]$$

$$= \left\{2\cos\varphi_0\cos 2\varphi_0 + 2\sin\varphi_0\sin 2\varphi_0\right\}\frac{P_2^2(\eta_0)P_2^2(\xi_0)}{h_\varphi}$$

$$= -18\left[\left(\eta_0^2-1\right)\left(1-\xi_0^2\right)\right]^{1/2}\frac{\cos\varphi_0}{c} \ = \ -\frac{18x_0}{c} \qquad \text{A3.46}$$

$$\text{The } \hat{j} \text{ component } = \left\{-3\sin\varphi_0\cos 2\varphi_0 + 2\cos\varphi_0\sin 2\varphi_0\right\}\frac{P_2^2(\eta_0)P_2^2(\xi_0)}{h_\varphi} +$$

$$\frac{c\sin\varphi_0\cos 2\varphi_0 P_2^2(\eta_0)P_2^2(\xi_0)}{h_\eta h_\xi}\left[\frac{5\xi_0^2}{\left(1-\xi_0^2\right)} - \frac{5\eta_0^2}{\left(1-\eta_0^2\right)}\right]$$

$$= 18\left[\left(\eta_0^2-1\right)\left(1-\xi_0^2\right)\right]^{1/2}\frac{\sin\varphi_0}{c} \ = \ \frac{18y_0}{c} \qquad \text{A3.47}$$

$$\text{The } \hat{k} \text{ component } = 0 \qquad \text{A3.48}$$

$$\text{Thus } A_{22} \ = \ -\frac{5}{16c^2}\left(x_0 D_x - y_0 D_y\right) \qquad \text{A3.49}$$

B_{21}

A3.50

$$\text{The } \hat{i} \text{ component } = 0$$

$$\text{The } \hat{j} \text{ component } = \frac{9z_0}{c}j \qquad \text{A3.51}$$

$$\text{The } \hat{k} \text{ component } = \frac{9y_0}{c}j \qquad \text{A3.52}$$

$$\text{Thus } B_{21} \ = \ -\frac{5}{2c^2}\left(z_0 D_y + y_0 D_y\right)j \qquad \text{A3.53}$$

B_{22}

The $\hat{i}$ component $= -\frac{18y_0}{c}$ A3.54

The $\hat{j}$ component $= -\frac{18x_0}{c}$ A3.55

The $\hat{k}$ component $= 0$ A3.56

Thus $B_{22} = -\frac{5}{16c^2}\left(y_0 D_x + x_0 D_y\right)$ A3.57

Appendix 3.2a Quadrupole Coefficients for the Prolate Spheroid

The infinite medium potential due to a quadrupole projected as aspheroid wave onto the surface η_e of a prolate spheroid is:

$$\Phi_{(\eta,\xi)} = \frac{1}{4\pi\sigma}\sum_{n=0}^{\infty}\sum_{m=0}^{n}\left(\mathcal{A}_{nm}\cos m\varphi + \mathcal{B}_{nm}\sin m\varphi\right)P_n^m(\eta_e)P_n^m(\xi) \quad \text{A3.58}$$

where the source at $(\eta_0, \xi_0, \varphi_0)$ is described in terms of the multipole coefficients:

$$\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix} = \frac{1}{c}\left(2-\delta_m^0\right)(-1)^m(2n+1)\left[\frac{(n-m)!}{(n+m)!}\right]^2 \cdot \left\{\nabla \cdot \tilde{Q} \cdot \nabla\left[P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right]\right\} \quad \text{A3.59}$$

The gradient of the source spatial configuration is expressed as:

$$\nabla\left[P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right] = \frac{1}{h_\eta}\left[P_n'^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right]\hat{\eta} + \frac{1}{h_\xi}\left[P_n^m(\eta_0)P_n'^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix}\right]\hat{\xi} + \frac{1}{h_\varphi}\left[P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}-m\sin m\varphi_0\\ m\cos m\varphi_0\end{Bmatrix}\right]\hat{\varphi} \quad \text{A3.60}$$

Let $\gamma_{nm} = \frac{1}{c}\left(2-\delta_m^0\right)(-1)^m(2n+1)\left[\frac{(n-m)!}{(n+m)!}\right]^2$ A3.61

For the quadrupole source term let:

$$\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix} = \nabla \cdot \tilde{Q} \cdot \nabla\left[\sum_{n=0}^{\infty}\sum_{m=0}^{n}\gamma_{nm}\overline{F}_{x,y,z}\right] \quad \text{A3.62}$$

where, taking the first ∇ operator, the $\hat{i}$ component is:

$$\bar{F}_x = \frac{H1}{c}\left[\eta_0 P_n'^m(\eta_0)P_n^m(\xi_0) - \xi_0 P_n^m(\eta_0)P_n'^m(\xi_0)\right]\begin{Bmatrix}\cos\varphi_0\cos m\varphi_0\\ \cos\varphi_0\sin m\varphi_0\end{Bmatrix} - \frac{H2}{c}P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}-m\sin\varphi_0\sin m\varphi_0\\ m\sin\varphi_0\cos m\varphi_0\end{Bmatrix} \quad \text{A3.63}$$

The $\hat{j}$ component is:

$$\bar{F}_y = \frac{H1}{c}\left[\eta_0 P_n'^m(\eta_0)P_n^m(\xi_0) - \xi_0 P_n^m(\eta_0)P_n'^m(\xi_0)\right]\begin{Bmatrix}\sin\varphi_0\cos m\varphi_0\\ \sin\varphi_0\sin m\varphi_0\end{Bmatrix} - \frac{H2}{c}P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}-m\cos\varphi_0\sin m\varphi_0\\ m\cos\varphi_0\cos m\varphi_0\end{Bmatrix} \quad \text{A3.64}$$

The $\hat{k}$ component is:

$$\bar{F}_z = \frac{H3}{c}(n-m+1)\left[\xi_0 P_{n+1}^m(\eta_0)P_n^m(\xi_0) - \eta_0 P_n^m(\eta_0)P_{n+1}^m(\xi_0)\right]\begin{Bmatrix}\cos m\varphi_0\\ \sin m\varphi_0\end{Bmatrix} \quad \text{A3.65}$$

and where:

$$H1 = \frac{\left(\eta_o^2-1\right)^{1/2}\left(1-\xi_o^2\right)^{1/2}}{\left(\eta_o^2-\xi_o^2\right)} \qquad H2 = \frac{1}{\left(\eta_o^2-1\right)^{1/2}\left(1-\xi_o^2\right)^{1/2}} \qquad H3 = \frac{1}{\left(\eta_o^2-\xi_o^2\right)} \quad \text{A3.66}$$

Taking the second ∇ operator there are 9 terms which will be represented as:

$$\bar{F}_{x\eta} = \frac{1}{h_\eta}\frac{\partial}{\partial\eta}(\bar{F}_x)\hat{\eta} \qquad \bar{F}_{x\xi} = \frac{1}{h_\xi}\frac{\partial}{\partial\xi}(\bar{F}_x)\hat{\xi} \qquad \bar{F}_{x\varphi} = \frac{1}{h_\varphi}\frac{\partial}{\partial\varphi}(\bar{F}_x)\hat{\varphi} \quad \text{A3.67}$$

$$\bar{F}_{y\eta} = \frac{1}{h_\eta}\frac{\partial}{\partial\eta}(\bar{F}_y)\hat{\eta} \quad \text{..... etc}$$

and $H1'_\eta = \frac{\partial}{\partial\eta}(H1)$ ..... etc

Some of the resulting terms are shown here as an example:

Two of the η terms:

$$\bar{F}_{x\eta} = \begin{bmatrix}\frac{H1'_\eta}{h_\eta c}\left[\eta_0 P_n'^m(\eta_0)P_n^m(\xi_0) - \xi_0 P_n^m(\eta_0)P_n'^m(\xi_0)\right] + \\ \frac{H1_\eta}{h_\eta c}\left[P_n'^m(\eta_0)P_n^m(\xi_0) + \eta_0 P_n''^m(\eta_0)P_n^m(\xi_0) - \right. \\ \left.\xi_0 P_n'^m(\eta_0)P_n'^m(\xi_0)\right]\end{bmatrix}\begin{Bmatrix}\cos\varphi_0\cos m\varphi_0\\ \cos\varphi_0\sin m\varphi_0\end{Bmatrix} - \left[\frac{H2'}{h_\eta c}P_n^m(\eta_0) + \frac{H2}{h_\eta c}P_n'^m(\eta_0)\right]P_n^m(\xi_0)\begin{Bmatrix}-m\sin\varphi_0\sin m\varphi_0\\ m\sin\varphi_0\cos m\varphi_0\end{Bmatrix} \quad \text{A3.68}$$

$$\bar{F}_{z\eta} = \begin{bmatrix} \frac{H3'}{h_\eta c}(n-m+1)\left[\xi_0 P_{n+1}^m(\eta_0)P_n^m(\xi_0) - \eta_0 P_n^m(\eta_0)P_{n+1}^m(\xi_0)\right] + \\ \frac{H3}{h_f}(n-m+1)\begin{bmatrix} \xi_0 P_{n+1}'^m(\eta_0)P_n^m(\xi_0) - \eta_0 P_n'^m(\eta_0)P_{n+1}^m(\xi_0) - \\ P_n^m(\eta_0)P_{n+1}^m(\xi_0) \end{bmatrix} \end{bmatrix} \begin{Bmatrix} \cos m\varphi_0 \\ \sin m\varphi_0 \end{Bmatrix} \quad \text{A3.69}$$

Two of the φ terms:

$$\bar{F}_{x\varphi} = \frac{H1}{h_\varphi c}\begin{bmatrix} \eta_0 P_n'^m(\eta_0)P_n^m(\xi_0) - \\ \xi_0 P_n^m(\eta_0)P_n'^m(\xi_0) \end{bmatrix} \begin{Bmatrix} -\sin\varphi_0 \cos m\varphi_0 - m\cos\varphi_0 \cos m\varphi_0 \\ -\sin\varphi_0 \sin m\varphi_0 + m\cos\varphi_0 \sin m\varphi_0 \end{Bmatrix} \quad \text{A3.70}$$

$$- \frac{H2}{c} P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix} -m\cos\varphi_0 \sin m\varphi_0 - m^2 \sin\varphi_0 \cos m\varphi_0 \\ -m\cos\varphi_0 \cos m\varphi_0 - m^2 \sin\varphi_0 \sin m\varphi_0 \end{Bmatrix}$$

$$\bar{F}_{z\varphi} = \frac{H3}{h_\varphi c}(n-m+1)\begin{bmatrix} \xi_0 P_{n+1}^m(\eta_0)P_n^m(\xi_0) - \\ \eta_0 P_n^m(\eta_0)P_{n+1}^m(\xi_0) \end{bmatrix} \begin{Bmatrix} -m\sin m\varphi_0 \\ m\cos m\varphi_0 \end{Bmatrix} \quad \text{A3.71}$$

and similarly for

$\bar{F}_{x\xi}, \bar{F}_{x\varphi}, \bar{F}_{y\eta}$ ..... etc

These combine:

for the Q_{xx} coefficient $\left(\frac{c\eta_0}{h_\xi}\bar{F}_{x\eta} - \frac{c\xi_0}{h_\eta}\bar{F}_{x\xi}\right)\cos\varphi_0 - \bar{F}_{x\varphi}\sin\varphi_0$ A3.72

for the Q_{xy} coefficient $\left(\frac{c\eta_0}{h_\xi}\bar{F}_{x\eta} - \frac{c\xi_0}{h_\eta}\bar{F}_{x\xi}\right)\sin\varphi_0 - \bar{F}_{x\varphi}\cos\varphi_0$ A3.73

for the Q_{xz} coefficient $\left(\frac{c\xi_0}{h_\eta}\bar{F}_{x\eta} + \frac{c\eta_0}{h_\xi}\bar{F}_{x\xi}\right)$ A3.74

..... etc for the remaining 6 components $Q_{yx}...Q_{zz}$

In the above:

$$H1'_\eta = \frac{\eta_0\left(1-\xi_o^2\right)^{1/2}}{\left(\eta_o^2-1\right)^{1/2}}\frac{\left[2-\left(\eta_o^2+\xi_o^2\right)^{1/2}\right]}{\left(\eta_o^2-\xi_o^2\right)} \qquad H1'_\xi = \frac{\xi_0\left(\eta_o^2-1\right)^{1/2}}{\left(1-\xi_o^2\right)^{1/2}}\frac{\left[2-\left(\eta_o^2+\xi_o^2\right)^{1/2}\right]}{\left(\eta_o^2-\xi_o^2\right)}$$

$$H2'_\eta = -\frac{\eta_0}{\left(\eta_o^2-1\right)^{1/2}\left(1-\xi_o^2\right)^{1/2}} \qquad H2'_\xi = \frac{\xi_0}{\left(\eta_o^2-1\right)^{1/2}\left(1-\xi_o^2\right)^{1/2}}$$

$$H3'_\eta = -\frac{2\eta_0}{\left(\eta_o^2-\xi_o^2\right)} \qquad H3'_\xi = \frac{2\xi_0}{\left(\eta_o^2-\xi_o^2\right)} \qquad \text{A3.75}$$

Component Evaluation

The $\mathcal{A}$ and $\mathcal{B}$ components are evaluated, as an example, only for n=2

at m=0

There are no $\mathcal{B}_{20}$ components only $\mathcal{A}_{20}$

$$\left\{\begin{matrix}\overline{F}_x\\ \overline{F}_y\end{matrix}\right\}\Bigg|_{n=2,m=0} = -\frac{3}{2c}\left(\eta_0^2-1\right)^{1/2}\left(1-\xi_0^2\right)^{1/2}\left\{\begin{matrix}\cos\varphi_0\\ \sin\varphi_0\end{matrix}\right\} \qquad \text{A3.76}$$

$$\overline{F}_z\Big|_{n=2,m=0} = \frac{3}{c}\xi_0\eta_0\cos\varphi_0 \qquad \text{A3.77}$$

$$\frac{1}{h_\eta}\left\{\begin{matrix}\overline{F}_{x\eta}\\ \overline{F}_{y\eta}\end{matrix}\right\}\Bigg|_{n=2,m=0} = -\frac{3\eta}{2c^2}\frac{\left(1-\xi_0^2\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}}\left\{\begin{matrix}\cos\varphi_0\\ \sin\varphi_0\end{matrix}\right\} \qquad \text{A3.78}$$

$$\frac{1}{h_\xi}\left\{\begin{matrix}\overline{F}_{x\xi}\\ \overline{F}_{y\xi}\end{matrix}\right\}\Bigg|_{n=2,m=0} = \frac{3\xi}{2c^2}\frac{\left(\eta_0^2-1\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}}\left\{\begin{matrix}\cos\varphi_0\\ \sin\varphi_0\end{matrix}\right\} \qquad \text{A3.79}$$

$$\frac{1}{h_\varphi}\left\{\begin{matrix}\overline{F}_{x\varphi}\\ \overline{F}_{y\varphi}\end{matrix}\right\}\Bigg|_{n=2,m=0} = -\frac{3}{2c^2}\left\{\begin{matrix}-\sin\varphi_0\\ \cos\varphi_0\end{matrix}\right\} \qquad \text{A3.80}$$

$$\frac{1}{h_\eta}\overline{F}_{z\eta}\Big|_{n=2,m=0} = \frac{3\xi}{c^2}\frac{\left(\eta_0^2-1\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}} \qquad \text{A3.81}$$

$$\overline{F}_{z\varphi}\Big|_{n=2,m=0} = 0 \qquad \text{A3.82}$$

$$\begin{Bmatrix} Q_{xx} \\ Q_{xy} \end{Bmatrix} = \left[\frac{c\eta}{h_\xi}\left(\frac{1}{h_\eta}\overline{F}_{x\eta}\right) - \frac{c\xi}{h_\eta}\left(\frac{1}{h_\xi}\overline{F}_{x\xi}\right) \right] \begin{Bmatrix} \cos\varphi_0 \\ \sin\varphi_0 \end{Bmatrix} + \left(\frac{1}{h_\varphi}\overline{F}_{x\varphi}\right) \begin{Bmatrix} -\sin\varphi_0 \\ \cos\varphi_0 \end{Bmatrix} \quad \text{A3.83}$$

$$= \begin{Bmatrix} -\frac{3}{2c^2} \\ 0 \end{Bmatrix}$$

$$Q_{xz} = \frac{c\xi}{h_\eta}\left(\frac{1}{h_\eta}\overline{F}_{x\eta}\right) + \frac{c\eta}{h_\xi}\left(\frac{1}{h_\xi}\overline{F}_{x\xi}\right) \quad = 0 \quad \text{A3.84}$$

$$\begin{Bmatrix} Q_{yx} \\ Q_{yy} \end{Bmatrix} = \left[\frac{c\eta}{h_\xi}\left(\frac{1}{h_\eta}\overline{F}_{y\eta}\right) - \frac{c\xi}{h_\eta}\left(\frac{1}{h_\xi}\overline{F}_{y\xi}\right) \right] \begin{Bmatrix} \cos\varphi_0 \\ \sin\varphi_0 \end{Bmatrix} + \left(\frac{1}{h_\varphi}\overline{F}_{y\varphi}\right) \begin{Bmatrix} -\sin\varphi_0 \\ \cos\varphi_0 \end{Bmatrix} \quad \text{A3.85}$$

$$= \begin{Bmatrix} 0 \\ -\frac{3}{2c^2} \end{Bmatrix}$$

$$Q_{yz} = \frac{c\xi}{h_\eta}\left(\frac{1}{h_\eta}\overline{F}_{y\eta}\right) + \frac{c\eta}{h_\xi}\left(\frac{1}{h_\xi}\overline{F}_{y\xi}\right) \quad = 0 \quad \text{A3.86}$$

$$\begin{Bmatrix} Q_{zx} \\ Q_{zy} \end{Bmatrix} = \left[\frac{c\eta}{h_\xi}\left(\frac{1}{h_\eta}\overline{F}_{z\eta}\right) - \frac{c\xi}{h_\eta}\left(\frac{1}{h_\xi}\overline{F}_{z\xi}\right) \right] \begin{Bmatrix} \cos\varphi_0 \\ \sin\varphi_0 \end{Bmatrix} + \left(\frac{1}{h_\varphi}\overline{F}_{z\varphi}\right) \begin{Bmatrix} -\sin\varphi_0 \\ \cos\varphi_0 \end{Bmatrix} \quad \text{A3.87}$$

$$= \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

$$Q_{zz} = \frac{c\xi}{h_\eta}\left(\frac{1}{h_\eta}\overline{F}_{z\eta}\right) + \frac{c\eta}{h_\xi}\left(\frac{1}{h_\xi}\overline{F}_{z\xi}\right) \quad = \frac{3}{c^2} \quad \text{A3.88}$$

$$\gamma_{2,0} = \frac{5}{c} \quad \text{A3.89}$$

$$\text{Thus } \mathcal{A}_{2,0} = \frac{15}{2c^3}\left(-Q_{xx} - Q_{yy} + 2Q_{zz}\right) \quad \text{A3.90}$$

<u>At m=1</u>

$$\overline{F}_x \begin{Bmatrix} \mathcal{A}_{nm} \\ \mathcal{B}_{nm} \end{Bmatrix}\Bigg|_{n=2,m=1} = \frac{9\eta_0\xi_0}{c} j \begin{Bmatrix} \cos^2\varphi_0 \\ \sin\varphi_0\cos\varphi_0 \end{Bmatrix} - \frac{9\eta_0\xi_0}{c} j \begin{Bmatrix} \sin^2\varphi_0 \\ \sin\varphi_0\cos\varphi_0 \end{Bmatrix} \quad \text{A3.91}$$

$$\overline{F}_y \begin{Bmatrix} \mathcal{A}_{nm} \\ \mathcal{B}_{nm} \end{Bmatrix}\Bigg|_{n=2,m=1} = \frac{9\eta_0\xi_0}{c} j \begin{Bmatrix} \sin\varphi_0\cos\varphi_0 \\ \sin^2\varphi_0 \end{Bmatrix} - \frac{9\eta_0\xi_0}{c} j \begin{Bmatrix} -\sin\varphi_0\cos\varphi_0 \\ \cos^2\varphi_0 \end{Bmatrix} \quad \text{A3.92}$$

$$\overline{F}_z \begin{Bmatrix} \mathcal{A}_{nm} \\ \mathcal{B}_{nm} \end{Bmatrix}\Bigg|_{n=2,m=1} = \frac{9}{c}\left(\eta_0^2 - 1\right)^{1/2}\left(1 - \xi_0^2\right)^{1/2} j \begin{Bmatrix} \sin\varphi_0 \\ \cos\varphi_0 \end{Bmatrix} \quad \text{A3.93}$$

where $j^2 = -1$

$$\frac{1}{h_\eta}\bar{F}_{x\eta}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = \frac{9\xi_0}{c^2}\frac{\left(\eta_0^2-1\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}} j\left\{\begin{matrix}1\\0\end{matrix}\right\} \qquad \text{A3.94}$$

$$\frac{1}{h_\xi}\bar{F}_{x\xi}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = \frac{9\eta_0}{c^2}\frac{\left(1-\xi_0^2\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}} j\left\{\begin{matrix}1\\0\end{matrix}\right\} \qquad \text{A3.95}$$

$$\frac{1}{h_\eta}\bar{F}_{y\eta}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = \frac{9\xi_0}{c^2}\frac{\left(\eta_0^2-1\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}} j\left\{\begin{matrix}0\\1\end{matrix}\right\} \qquad \text{A3.96}$$

$$\frac{1}{h_\xi}\bar{F}_{x\xi}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = \frac{9\eta_0}{c^2}\frac{\left(1-\xi_0^2\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}} j\left\{\begin{matrix}0\\1\end{matrix}\right\} \qquad \text{A3.97}$$

$$\frac{1}{h_\varphi}\bar{F}_{x\varphi}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = \frac{9\eta_0\xi_0}{c^2\left(\eta_0^2-1\right)^{1/2}\left(1-\xi_0^2\right)^{1/2}} j\left\{\begin{matrix}-2\sin\varphi_0\cos\varphi_0\\ -\sin^2\varphi_0+\cos^2\varphi_0\end{matrix}\right\} -$$

$$\frac{9\eta_0\xi_0}{c^2\left(\eta_0^2-1\right)^{1/2}\left(1-\xi_0^2\right)^{1/2}} j\left\{\begin{matrix}-2\sin\varphi_0\cos\varphi_0\\ -\sin^2\varphi_0+\cos^2\varphi_0\end{matrix}\right\} = \left\{\begin{matrix}0\\0\end{matrix}\right\} \qquad \text{A3.98}$$

$$\frac{1}{h_\varphi}\bar{F}_{y\varphi}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = \frac{9\eta_0\xi_0}{c^2\left(\eta_0^2-1\right)^{1/2}\left(1-\xi_0^2\right)^{1/2}} j\left\{\begin{matrix}-\sin^2\varphi_0+\cos^2\varphi_0\\ 2\sin\varphi_0\cos\varphi_0\end{matrix}\right\} -$$

$$\frac{9\eta_0\xi_0}{c^2\left(\eta_0^2-1\right)^{1/2}\left(1-\xi_0^2\right)^{1/2}} j\left\{\begin{matrix}-\sin^2\varphi_0+\cos^2\varphi_0\\ 2\sin\varphi_0\cos\varphi_0\end{matrix}\right\} = \left\{\begin{matrix}0\\0\end{matrix}\right\} \qquad \text{A3.99}$$

$$\frac{1}{h_\eta}\bar{F}_{z\eta}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = \frac{9\eta_0}{c^2}\frac{\left(1-\xi_0^2\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}} j\left\{\begin{matrix}\cos\varphi_0\\ \sin\varphi_0\end{matrix}\right\} \qquad \text{A3.100}$$

$$\frac{1}{h_\xi}\bar{F}_{z\xi}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = -\frac{9\xi_0}{c^2}\frac{\left(\eta_0^2-1\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}} j\left\{\begin{matrix}\cos\varphi_0\\ \sin\varphi_0\end{matrix}\right\} \qquad \text{A3.101}$$

$$\frac{1}{h_\varphi}\bar{F}_{z\varphi}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = -\frac{9}{c^2} j\left\{\begin{matrix}-\sin\varphi_0\\ \cos\varphi_0\end{matrix}\right\} \qquad \text{A3.102}$$

$$Q_{xx}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = Q_{xy}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Bigg|_{n=2,m=1} = \left\{\begin{matrix}0\\0\end{matrix}\right\} \qquad \text{A3.103}$$

$$Q_{yx}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=1} = Q_{yy}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=1} = \left\{\begin{matrix}0\\ 0\end{matrix}\right\} \qquad \text{A3.104}$$

$$Q_{zx}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=1} = \left\{\begin{matrix}\frac{9}{c^2}j\\ 0\end{matrix}\right\} \quad Q_{zy}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=1} = \left\{\begin{matrix}0\\ \frac{9}{c^2}j\end{matrix}\right\} \qquad \text{A3.105}$$

$$Q_{xz}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=1} = \left\{\begin{matrix}\frac{9}{c^2}j\\ 0\end{matrix}\right\} \quad Q_{yz}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=1} = \left\{\begin{matrix}0\\ \frac{9}{c^2}j\end{matrix}\right\} \qquad \text{A3.106}$$

$$Q_{zz}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=1} = \left\{\begin{matrix}0\\ 0\end{matrix}\right\} \qquad \text{A3.107}$$

$$\gamma\Big|_{n=2,m=1} = -\frac{5}{18c} \qquad \text{A3.108}$$

$$\text{Thus } \mathcal{A}_{2,1} = -\frac{5}{2c^3}j(Q_{xz}+Q_{zx}) \qquad \text{A3.109}$$

$$\text{and } \mathcal{B}_{2,1} = -\frac{5}{2c^3}j(Q_{zy}+Q_{yz}) \qquad \text{A3.110}$$

<u>At m=2</u>

$$\bar{F}_x\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=2} = -\frac{18}{c}\left(\eta_0^2-1\right)^{1/2}\left(1-\xi_0^2\right)^{1/2}\left\{\begin{matrix}\cos\varphi_0\\ \sin\varphi_0\end{matrix}\right\} \qquad \text{A3.111}$$

$$\bar{F}_y\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=2} = -\frac{18}{c}\left(\eta_0^2-1\right)^{1/2}\left(1-\xi_0^2\right)^{1/2}\left\{\begin{matrix}-\sin\varphi_0\\ \cos\varphi_0\end{matrix}\right\} \qquad \text{A3.112}$$

$$\bar{F}_z\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=2} = \left\{\begin{matrix}0\\ 0\end{matrix}\right\} \qquad \text{A3.113}$$

$$\frac{1}{h_\eta}\bar{F}_{x\eta}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=2} = -\frac{18\eta_0}{c^2}\frac{\left(1-\xi_0^2\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}}\left\{\begin{matrix}\cos\varphi_0\\ \sin\varphi_0\end{matrix}\right\} \qquad \text{A3.114}$$

$$\frac{1}{h_\eta}\bar{F}_{y\eta}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=2} = -\frac{18\eta_0}{c^2}\frac{\left(1-\xi_0^2\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}}\left\{\begin{matrix}-\sin\varphi_0\\ \cos\varphi_0\end{matrix}\right\} \qquad \text{A3.115}$$

$$\frac{1}{h_\xi}\bar{F}_{x\xi}\left\{\begin{matrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{matrix}\right\}\Big|_{n=2,m=2} = \frac{18\xi_0}{c^2}\frac{\left(\eta_0^2-1\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}}\left\{\begin{matrix}\cos\varphi_0\\ \sin\varphi_0\end{matrix}\right\} \qquad \text{A3.116}$$

$$\frac{1}{h_\xi}\bar{F}_{y\xi}\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix}\Bigg|_{n=2,m=2} = \frac{18\xi_0}{c^2}\frac{\left(\eta_0^2-1\right)^{1/2}}{\left(\eta_0^2-\xi_0^2\right)^{1/2}}\begin{Bmatrix}-\sin\varphi_0\\ \cos\varphi_0\end{Bmatrix} \qquad \text{A3.117}$$

$$Q_{xx}\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix}\Bigg|_{n=2,m=2} = \begin{Bmatrix}-\frac{18}{c^2}\\ 0\end{Bmatrix} \quad Q_{xy}\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix}\Bigg|_{n=2,m=2} = \begin{Bmatrix}0\\ -\frac{18}{c^2}\end{Bmatrix} \qquad \text{A3.118}$$

$$Q_{yx}\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix}\Bigg|_{n=2,m=2} = \begin{Bmatrix}0\\ -\frac{18}{c^2}\end{Bmatrix} \quad Q_{yy}\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix}\Bigg|_{n=2,m=2} = \begin{Bmatrix}\frac{18}{c^2}\\ 0\end{Bmatrix} \qquad \text{A3.119}$$

$$Q_{xz}\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix}\Bigg|_{n=2,m=2} = Q_{yz}\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix}\Bigg|_{n=2,m=2} = \begin{Bmatrix}0\\ 0\end{Bmatrix} \qquad \text{A3.120}$$

$$Q_{zx}\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix}\Bigg|_{n=2,m=2} = Q_{zy}\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix}\Bigg|_{n=2,m=2} = Q_{zz}\begin{Bmatrix}\mathcal{A}_{nm}\\ \mathcal{B}_{nm}\end{Bmatrix}\Bigg|_{n=2,m=2} = \begin{Bmatrix}0\\ 0\end{Bmatrix} \qquad \text{A3.121}$$

$$\gamma\Big|_{n=2,m=2} = \frac{5}{288c} \qquad \text{A3.122}$$

Thus $\mathcal{A}_{2,2} = -\frac{1}{8c^3}\frac{5}{2}\left(Q_{xx}-Q_{yy}\right)$ A3.123

and $\mathcal{B}_{2,2} = -\frac{1}{8c^3}\frac{5}{2}\left(Q_{xy}+Q_{yx}\right)$ A3.124

Appendix 3.3 The Integration with respect to φ

Using only the terms relevant to the integration:

$$\bar{B}_v = \left(-\frac{1}{4\pi\sigma}\right)\left(-\frac{\mu\sigma}{4\pi}\right)\int_{\xi=-1}^{1}\int_{\varphi=0}^{2\pi}\sum_{n=0}^{\infty}\sum_{m=0}^{n}\alpha_{nm}P_n^m(\xi)\begin{Bmatrix}A_{nm}\cos m\varphi\\ B_{nm}\sin m\varphi\end{Bmatrix}\cdot$$

$$\sum_{n'=0}^{\infty}\sum_{m'=0}^{n'}\beta_{n',m'}\begin{vmatrix}\lambda_i\hat{i}\\ \lambda_j\hat{j}\\ \lambda_k\hat{k}\end{vmatrix}\,d\varphi d\xi \qquad \text{A3.125}$$

where

$$\alpha_{nm} = (-1)^m\frac{(n-m)!}{(n+m)!}\frac{1}{\eta_e^2-1}\frac{1}{P_n'^m(\eta_e)} \qquad \text{A3.126}$$

$$\beta_{n',m'} = \frac{1}{c}\left(2-\delta_{m'}^0\right)(-1)^{m'}(2n'+1)\left[\frac{(n'-m')!}{(n'+m')!}\right]^2\cdot Q_{n'}^{m'}(\eta_e)P_{n'}^{m'}(\xi_e)P_{n'}^{m'}(\eta_e)P_{n'}^{m'}(\xi) \qquad \text{A3.127}$$

$$\lambda_i = -cm'\xi\frac{h_\xi}{h_\eta}P_{n'}^{m'}(\xi)(\sin m'\varphi_e\cos m'\varphi-\cos m'\varphi_e\sin m'\varphi)\cos\varphi$$

$$+ \qquad \text{A3.128}$$

$$h_\varphi P_{n'}'^{m'}(\xi)(\cos m'\varphi_e\cos m'\varphi-\sin m'\varphi_e\sin m'\varphi)\sin\varphi$$

$$\lambda_j = -cm'\xi\frac{h_\xi}{h_\eta}P_{n'}^{m'}(\xi)(\sin m'\varphi_e\cos m'\varphi-\cos m'\varphi_e\sin m'\varphi)\sin\varphi$$

$$- \qquad \text{A3.129}$$

$$h_\varphi P_{n'}'^{m'}(\xi)(\cos m'\varphi_e\cos m'\varphi-\sin m'\varphi_e\sin m'\varphi)\cos\varphi$$

$$\lambda_k = cm'\eta_e P_{n'}^{m'}(\xi)(\sin m'\varphi_e\cos m'\varphi-\cos m'\varphi_e\sin m'\varphi) \qquad \text{A3.130}$$

Combining the λ's with cos mφ and sin mφ, it is noted that the integral is zero for the combinations of (cos cos sin) and (sin sin sin).

Otherwise there are 4 combinations for the i and j terms:

1] sin φsin m'φcos mφ 2] sin φcos m'φsin mφ

3] cos φcos m'φcos mφ 4] cos φsin m'φsin mφ

These integrals evaluate as follows:

	$m'=1 \;\; m=0$	$m'=0 \;\; m=1$	$m'=m+1$ $\sum_{m=1}^{n'}$	$m'=m-1$ $\sum_{m=2}^{n'}$
1]				
$\cos(m-1+m')\varphi$	2π	2π	-	-
$\cos(m+1-m')\varphi$	2π	-	2π	-
$-\cos(m-1-m')\varphi$	-	-2π	-	-2π
$-\cos(m+1+m')\varphi$	-	-	-	-
	---	---	---	---
	4π	-	$2\pi\delta_m^{m'-1}$ $m>1$	$-2\pi\delta_m^{m'+1}$ $m'>1$
2]				
$\cos(1-m'+m)\varphi$	2π	-	2π	-
$\cos(1+m'-m)\varphi$	-	2π	-	2π
$\cos(1-m'-m)\varphi$	2π	2π	-	-
$\cos(1+m'+m)\varphi$	-	-	-	-
	---	---	---	---
	4π	4π	$2\pi\delta_m^{m'-1}$ $m>1$	$2\pi\delta_m^{m'+1}$ $m'>1$
3]				
$\cos(1-m'-m)\varphi$	2π	2π	-	-
$\cos(1+m'-m)\varphi$	-	2π	-	2π
$-\cos(1-m'+m)\varphi$	-2π	-	-2π	-
$-\cos(1+m'+m)\varphi$	-	-	-	-
	---	---	---	---
	-	4π	$-2\pi\delta_m^{m'-1}$ $m>1$	$2\pi\delta_m^{m'+1}$ $m'>1$

4]				
$\cos(m'-1-m)\varphi$	2π	-	2π	-
$\cos(m'+1-m)\varphi$	-	2π	-	2π
$-\cos(m'-1+m)\varphi$	-2π	-2π	-	-
$-\cos(m'+1+m)\varphi$	-	-	-	-
	---	---	---	---
	-	-	$2\pi\delta_m^{m'-1}$	$2\pi\delta_m^{m'+1}$
			$m>1$	$m'>1$

Note: From the trigonometric identities, each result above is divided by 4 before substitution back into the main equation.

For the k terms there are 2 variations:

1] $\cos m'\varphi \cos m\varphi$ 2] $\sin m'\varphi \sin m\varphi$

	$m'=m=0$	$m'=m \quad m>0$
1]		
$\cos(m'-m)\varphi$	2π	2π
$\cos(m'+m)\varphi$	2π	-
	---	---
	4π	$2\pi\delta_m^{m'}$
2]		
$\cos(m'-m)\varphi$	2π	2π
$-\cos(m'+m)\varphi$	2π	-
	---	---
	-	$2\pi\delta_m^{m'}$

Note: From the trigonometric identities, each result above is divided by 2 before substitution back into the main equation.

Combining the results for the λ_i and λ_j terms:

$m' = 0 \quad m = 1$

$$A_{n1}\lambda_j\alpha_{n1}P_n^1(\xi)\beta_{n'0} = -\pi h_\varphi P_n^1(\xi)P_n'^0(\xi)\alpha_{n1}\beta_{n'0}A_{n1}\hat{j} \qquad \text{A3.131}$$

$$B_{n1}\lambda_i\alpha_{n1}P_n^1(\xi)\beta_{n'0} = \pi h_\varphi P_n^1(\xi)P_n'^0(\xi)\alpha_{n1}\beta_{n'0}B_{n1}\hat{i} \qquad \text{A3.132}$$

$m' = 1 \quad m = 0$

$$A_{n0}\lambda_i\alpha_{n0}P_n^0(\xi)\beta_{n'1} = \pi\sin\varphi_e P_n^0(\xi)\left(-c\xi\frac{h_\xi}{h_\eta}P_n^1(\xi) + h_\varphi P_n'^1(\xi)\right)\alpha_{n0}\beta_{n'1}A_{n0}\hat{i}$$

A3.133

$$A_{n0}\lambda_j\alpha_{n0}P_n^0(\xi)\beta_{n'1} = \pi\sin\varphi_e P_n^0(\xi)\left(c\xi\frac{h_\xi}{h_\eta}P_n^1(\xi) - h_\varphi P_n'^1(\xi)\right)\alpha_{n0}\beta_{n'1}A_{n0}\hat{j}$$

A3.134

For the A_{nm} multipole coefficients:

$m' = m + 1$

$$A_{nm}\lambda_i\alpha_{nm}P_n^m(\xi)\beta_{n',m+1} = \frac{\pi}{2}\sin(m+1)\varphi_e P_n^m(\xi)\begin{pmatrix} -c\xi\dfrac{h_\xi}{h_\eta}(m+1)P_n^{m+1}(\xi) + \\ h_\varphi P_n'^{m+1}(\xi) \end{pmatrix}\cdot$$

$$\alpha_{nm}\beta_{n',m+1}A_{nm}\hat{i}$$

A3.135

$$A_{nm}\lambda_j\alpha_{nm}P_n^m(\xi)\beta_{n',m+1} = \frac{\pi}{2}\cos(m+1)\varphi_e P_n^m(\xi)\begin{pmatrix} c\xi\dfrac{h_\xi}{h_\eta}(m+1)P_n^{m+1}(\xi) - \\ h_\varphi P_n'^{m+1}(\xi) \end{pmatrix}\cdot$$

$$\alpha_{nm}\beta_{n',m+1}A_{nm}\hat{j}$$

A3.136

which combined becomes:

$$A_{nm}\begin{bmatrix}\lambda_i\\ \lambda_j\end{bmatrix}\alpha_{nm}P_n^m(\xi)\beta_{n',m+1} = \frac{\pi}{2}\begin{bmatrix}-\sin(m+1)\varphi_e\\ \cos(m+1)\varphi_e\end{bmatrix}P_n^m(\xi)\begin{pmatrix} c\xi\frac{h_\xi}{h_\eta}(m+1)P_n^{m+1}(\xi) - \\ h_\varphi P_n'^{m+1}(\xi)\end{pmatrix}.$$

$$\alpha_{nm}\beta_{n',m+1}A_{nm}\begin{bmatrix}\hat{i}\\ \hat{j}\end{bmatrix}$$

A3.137

and for $m' = m - 1$

$$A_{nm}\begin{bmatrix}\lambda_i\\ \lambda_j\end{bmatrix}\alpha_{nm}P_n^m(\xi)\beta_{n',m-1} = \frac{\pi}{2}\begin{bmatrix}-\sin(m-1)\varphi_e\\ -\cos(m-1)\varphi_e\end{bmatrix}P_n^m(\xi)\begin{pmatrix} c\xi\frac{h_\xi}{h_\eta}(m-1)P_n^{m-1}(\xi) + \\ h_\varphi P_n'^{m-1}(\xi)\end{pmatrix}.$$

$$\alpha_{nm}\beta_{n',m-1}A_{nm}\begin{bmatrix}\hat{i}\\ \hat{j}\end{bmatrix}$$

A3.138

For the B_{nm} multipole coefficients:

$m' = m + 1$

$$B_{nm}\begin{bmatrix}\lambda_i\\ \lambda_j\end{bmatrix}\alpha_{nm}P_n^m(\xi)\beta_{n',m+1} = \frac{\pi}{2}\begin{bmatrix}\cos(m+1)\varphi_e\\ \sin(m+1)\varphi_e\end{bmatrix}P_n^m(\xi)\begin{pmatrix} c\xi\frac{h_\xi}{h_\eta}(m+1)P_n^{m+1}(\xi) - \\ h_\varphi P_n'^{m+1}(\xi)\end{pmatrix}.$$

$$\alpha_{nm}\beta_{n',m+1}B_{nm}\begin{bmatrix}\hat{i}\\ \hat{j}\end{bmatrix}$$

A3.139

and for $m' = m - 1$

$$B_{nm}\begin{bmatrix}\lambda_i\\ \lambda_j\end{bmatrix}\alpha_{nm}P_n^m(\xi)\beta_{n',m-1} = \frac{\pi}{2}\begin{bmatrix}\cos(m-1)\varphi_e\\ -\sin(m-1)\varphi_e\end{bmatrix}P_n^m(\xi)\begin{pmatrix} c\xi\frac{h_\xi}{h_\eta}(m-1)P_n^{m-1}(\xi) + \\ h_\varphi P_n'^{m-1}(\xi)\end{pmatrix}.$$

$$\alpha_{nm}\beta_{n',m-1}B_{nm}\begin{bmatrix}\hat{i}\\ \hat{j}\end{bmatrix}$$

A3.140

The $\hat{k}$ components

There are no λ_k components for $m = m' = 0$

For $m = m' > 0$

$$\lambda_k \begin{bmatrix} A_{nm} \\ B_{nm} \end{bmatrix} \alpha_{nm} P_n^m(\xi) \beta_{n',m} = \pi \begin{bmatrix} \sin m\varphi_e \\ -\cos m\varphi_e \end{bmatrix} cm\eta_e P_n^m(\xi) P_{n'}^m(\xi) \alpha_{nm} \beta_{n',m} \begin{bmatrix} A_{nm} \\ B_{nm} \end{bmatrix} \hat{k} \qquad \text{A3.141}$$

Appendix 3.4 The Integration with respect to ξ

Use is made of the following recursion identities for the Associated Legendre polynomials:

$$P_{n'}^m(\xi) = \frac{\xi(m-1)}{\left(1-\xi^2\right)^{1/2}} P_{n'}^{m-1}(\xi) + \left(1-\xi^2\right)^{1/2} P_{n'}^{m-1}(\xi) \qquad \text{A3.142}$$

$$(n'-m)(n'+m+1) P_{n'}^m(\xi) = \frac{\xi(m-1)}{\left(1-\xi^2\right)^{1/2}} P_{n'}^{m+1}(\xi) - \left(1-\xi^2\right)^{1/2} P_{n'}^{m+1}(\xi) \qquad \text{A3.143}$$

$$\int_{-1}^{1} P_n^m(\xi) P_{n'}^m(\xi) d\xi = \frac{2}{(2n+1)} \frac{(n+m)!}{(n-m)!} \delta_n^{n'} \qquad \text{A3.143}$$

For the cases:

$m' = 0 \quad m = 1$

$$A_{n1} \lambda_j \alpha_{n1} P_n^1(\xi) \beta_{n'0} = -\pi c \left(\eta_e^2 - 1\right)^{1/2} \frac{2}{2n+1} \alpha_{n1} \beta_{n0} A_{n1} \hat{j} \qquad \text{A3.144}$$

$$B_{n1} \lambda_j \alpha_{n1} P_n^1(\xi) \beta_{n'0} = \pi c \left(\eta_e^2 - 1\right)^{1/2} \frac{2}{2n+1} \alpha_{n1} \beta_{n0} B_{n1} \hat{i} \qquad \text{A3.145}$$

$m' = 1 \quad m = 0$

$$A_{n0} \lambda_i \alpha_{n0} P_n^0(\xi) \beta_{n'1} = \pi \sin \varphi_e c \left(\eta_e^2 - 1\right)^{1/2} \frac{2n(n+1)}{2n+1} \alpha_{n0} \beta_{n1} A_{n0} \hat{i} \qquad \text{A3.146}$$

$$A_{n0} \lambda_j \alpha_{n0} P_n^0(\xi) \beta_{n'1} = \pi \cos \varphi_e c \left(\eta_e^2 - 1\right)^{1/2} \frac{2n(n+1)}{2n+1} \alpha_{n0} \beta_{n1} A_{n0} \hat{j} \qquad \text{A3.147}$$

for $m' = m+1$

$$\sum_{m=1}^{n} A_{nm}\begin{bmatrix}\lambda_i\\ \lambda_j\end{bmatrix}\alpha_{nm}P_n^m(\xi)\beta_{n',m+1} = \frac{\pi}{2}\begin{bmatrix}-\sin(m+1)\varphi_e\\ \cos(m+1)\varphi_e\end{bmatrix}c\left(\eta_e^2-1\right)^{1/2}\frac{2}{2n+1}\frac{(n+m)!}{(n-m)!}\cdot$$

$$(n-m)(n+m+1)\alpha_{nm}\beta_{n,m+1}A_{nm}\begin{bmatrix}\hat{i}\\ \hat{j}\end{bmatrix}$$

A3.148

$$\sum_{m=1}^{n} B_{nm}\begin{bmatrix}\lambda_i\\ \lambda_j\end{bmatrix}\alpha_{nm}P_n^m(\xi)\beta_{n',m+1} = \frac{\pi}{2}\begin{bmatrix}\cos(m+1)\varphi_e\\ \sin(m+1)\varphi_e\end{bmatrix}c\left(\eta_e^2-1\right)^{1/2}\frac{2}{2n+1}\frac{(n+m)!}{(n-m)!}\cdot$$

$$(n-m)(n+m+1)\alpha_{nm}\beta_{n,m+1}B_{nm}\begin{bmatrix}\hat{i}\\ \hat{j}\end{bmatrix}$$

A3.149

This is equivalent to a summation from m=2 plus a term evaluated at m=1:

$$\frac{\pi}{2}\begin{bmatrix}-\sin 2\varphi_e\\ \cos 2\varphi_e\end{bmatrix}c\left(\eta_e^2-1\right)^{1/2}\frac{2}{2n+1}\frac{(n+1)!}{(n-1)!}\cdot$$

A3.150

$$(n-1)(n+2)\alpha_{n1}\beta_{n,2}A_{n1}\begin{bmatrix}\hat{i}\\ \hat{j}\end{bmatrix}$$

where $\alpha_{n1} = (-1)\frac{1}{\left(\eta_e^2-1\right)}\frac{(n+1)!}{(n-1)!}\frac{1}{P_n'^1(\eta_e)}$ A3.151

and $\beta_{n2} = \frac{2}{c}(-1)^2(2n+1)\left[\frac{(n-2)!}{(n+2)!}\right]^2\cdot Q_n^2(\eta_e)P_n^2(\xi_e)P_n^2(\eta_e)$ A3.152

which results in:

$$= -\frac{2\pi}{\left(\eta_e^2-1\right)^{1/2}}\frac{Q_n^2(\eta_e)P_n^2(\xi_e)P_n^2(\eta_e)}{P_n'^1(\eta_e)}\frac{1}{(n-1)(n+2)}\begin{bmatrix}-A_{n1}\sin 2\varphi_e + B_{n1}\cos 2\varphi_e\\ A_{n1}\cos 2\varphi_e + B_{n1}\sin 2\varphi_e\end{bmatrix}\begin{bmatrix}\hat{i}\\ \hat{j}\end{bmatrix}$$

A3.153

for $m' = m-1$

$$\sum_{m=2}^{n} A_{nm}\begin{bmatrix}\lambda_i\\ \lambda_j\end{bmatrix}\alpha_{nm}P_n^m(\xi)\beta_{n',m-1} = \frac{\pi}{2}\begin{bmatrix}-\sin(m-1)\varphi_e\\ \cos(m-1)\varphi_e\end{bmatrix}c\left(\eta_e^2-1\right)^{1/2}\frac{2}{2n+1}\frac{(n+m)!}{(n-m)!}\cdot$$

$$\alpha_{nm}\beta_{n,m-1}A_{nm}\begin{bmatrix}\hat{i}\\ \hat{j}\end{bmatrix}$$

A3.154

$$\sum_{m=2}^{n} B_{nm} \begin{bmatrix} \lambda_i \\ \lambda_j \end{bmatrix} \alpha_{nm} P_n^m(\xi) \beta_{n',m-1} = \frac{\pi}{2} \begin{bmatrix} \cos(m-1)\varphi_e \\ \sin(m-1)\varphi_e \end{bmatrix} c\left(\eta_e^2 - 1\right)^{1/2} \frac{2}{2n+1} \frac{(n+m)!}{(n-m)!} \cdot$$

$$\alpha_{nm} \beta_{n,m-1} B_{nm} \begin{bmatrix} \hat{i} \\ \hat{j} \end{bmatrix}$$

A3.155

In summary

All terms are $\sum_{n=1}^{\infty}$:

- the m=0 and m=1 cases:

$$\frac{2\pi}{\left(\eta_e^2 - 1\right)^{1/2}} \left\{ -2\sin\varphi_e \beta_{n0}^{*} A_{n0} - \beta_{n1}^{**} B_{n1} \right\} \hat{i} \qquad \text{A3.156}$$

$$\frac{2\pi}{\left(\eta_e^2 - 1\right)^{1/2}} \left\{ -2\cos\varphi_e \beta_{n0}^{*} A_{n0} - \beta_{n1}^{**} A_{n1} \right\} \hat{j} \qquad \text{A3.157}$$

- the additional m=1 case

$$-\frac{2\pi}{\left(\eta_e^2 - 1\right)^{1/2}} \beta_{n1}^{*} \begin{bmatrix} -A_{n1}\sin 2\varphi_e - B_{n1}\cos 2\varphi_e \\ A_{n1}\cos 2\varphi_e - B_{n1}\sin 2\varphi_e \end{bmatrix} \begin{bmatrix} \hat{i} \\ \hat{j} \end{bmatrix} \qquad \text{A3.158}$$

- plus the terms for m≥2

for the i component

$$\sum_{m=2}^{n} -\frac{2\pi}{\left(\eta_e^2 - 1\right)^{1/2}} \begin{bmatrix} A_{nm} \\ B_{nm} \end{bmatrix} \begin{bmatrix} -\sin(m+1)\varphi_e \beta_{nm}^{*} - \sin(m-1)\varphi_e \beta_{nm}^{**} \\ \cos(m+1)\varphi_e \beta_{nm}^{*} + \cos(m-1)\varphi_e \beta_{nm}^{**} \end{bmatrix} \hat{i} \qquad \text{A3.159}$$

for the j component

$$\sum_{m=2}^{n} -\frac{2\pi}{\left(\eta_e^2 - 1\right)^{1/2}} \begin{bmatrix} A_{nm} \\ B_{nm} \end{bmatrix} \begin{bmatrix} -\cos(m+1)\varphi_e \beta_{nm}^{*} - \cos(m-1)\varphi_e \beta_{nm}^{**} \\ \sin(m+1)\varphi_e \beta_{nm}^{*} - \sin(m-1)\varphi_e \beta_{nm}^{**} \end{bmatrix} \hat{j} \qquad \text{A3.160}$$

where:

$$\beta_{nm}^{*} = \frac{Q_n^{m+1}(\eta_e) P_n^{m+1}(\xi_e) P_n^{m+1}(\eta_e)}{(n+m+1)(n-m) P_n'^{m}(\eta_e)} \qquad \text{A3.161}$$

$$\beta_{nm}^{**} = \frac{Q_n^{m-1}(\eta_e)P_n^{m-1}(\xi_e)P_n^{m-1}(\eta_e)}{P_n'^m(\eta_e)}\left[(n-m+1)(n+m)\right]^2 \qquad \text{A3.162}$$

Note that for β^* the case of $n = m$ does not exist

These are combined to make a set of terms for m=0 plus a set for m≥1:

$$-\frac{4\pi}{\left(\eta_e^2-1\right)^{1/2}}\beta_{n0}^{*}\begin{bmatrix}\sin\varphi_e\\ \cos\varphi_e\end{bmatrix}\begin{bmatrix}\hat{i}\\ \hat{j}\end{bmatrix} -$$

$$\sum_{m=1}^{n}\frac{2\pi}{\left(\eta_e^2-1\right)^{1/2}}\begin{bmatrix}A_{nm}\\ B_{nm}\end{bmatrix}\begin{bmatrix}\left\{-\sin(m+1)\varphi_e\beta_{nm}^{*}-\sin(m-1)\varphi_e\beta_{nm}^{**}\right\}\hat{i}\\ \left\{\cos(m+1)\varphi_e\beta_{nm}^{*}+\cos(m-1)\varphi_e\beta_{nm}^{**}\right\}\\ \left\{-\cos(m+1)\varphi_e\beta_{nm}^{*}-\cos(m-1)\varphi_e\beta_{nm}^{**}\right\}\hat{j}\\ \left\{\sin(m+1)\varphi_e\beta_{nm}^{*}-\sin(m-1)\varphi_e\beta_{nm}^{**}\right\}\end{bmatrix} \qquad \text{A3.163}$$

For the k component

$$\frac{4\pi m\eta_e}{\left(\eta_e^2-1\right)}\begin{bmatrix}A_{nm}\\ B_{nm}\end{bmatrix}\begin{bmatrix}\sin m\varphi_e\\ -\cos m\varphi_e\end{bmatrix}\frac{Q_n^m(\eta_e)P_n^m(\xi_e)P_n^m(\eta_e)}{P_n'^m(\eta_e)}\hat{k} \qquad \text{A3.164}$$

Note that for all Legendre polynomials n≥m

Appendix 3.5 Reconciliation between this Thesis and [Cuffin and Cohen:1977]

This thesis result for the magnet field on the surface of a prolate spheroid:

$$\overline{B}_{v(\eta_e,\xi)} = -\frac{\mu}{4\pi}\frac{1}{\left(\eta_e^2-1\right)^{1/2}}\sum_{n=1}^{\infty}\beta_{n0}^{*}A_{n0}\begin{bmatrix}\sin\varphi_e\ \hat{i}\\ \cos\varphi_e\ \hat{j}\end{bmatrix} +$$

$$\sum_{m=1}^{n}\begin{Bmatrix}A_{nm}\\ B_{nm}\end{Bmatrix}\begin{bmatrix}\frac{1}{2}\left\{-\sin(m+1)\varphi_e\beta_{nm}^{*}-\sin(m-1)\varphi_e\beta_{nm}^{**}\right\}\hat{i}\\ \frac{1}{2}\left\{\cos(m+1)\varphi_e\beta_{nm}^{*}+\cos(m-1)\varphi_e\beta_{nm}^{**}\right\}\\ \frac{1}{2}\left\{\cos(m+1)\varphi_e\beta_{nm}^{*}-\cos(m-1)\varphi_e\beta_{nm}^{**}\right\}\hat{j}\\ \frac{1}{2}\left\{\sin(m+1)\varphi_e\beta_{nm}^{*}-\sin(m-1)\varphi_e\beta_{nm}^{**}\right\}\\ \left(-\frac{m\eta_eQ_n^m(\eta_e)P_n^m(\xi_e)P_n^m(\eta_e)}{\left(\eta_e^2-1\right)^{1/2}P_n'^m(\eta_e)}\right)\begin{Bmatrix}\sin m\varphi_e\\ -\cos m\varphi_e\end{Bmatrix}\hat{k}\end{bmatrix} \qquad \text{A3.165}$$

where

$$\beta^{*}_{nm} = \frac{Q_n^{m+1}(\eta_e)P_n^{m+1}(\xi_e)P_n^{m+1}(\eta_e)}{(n+m+1)(n-m)P_n'^{m}(\eta_e)}$$

$$\beta^{**}_{nm} = \frac{Q_n^{m-1}(\eta_e)P_n^{m-1}(\xi_e)P_n^{m-1}(\eta_e)}{P_n'^{m}(\eta_e)}\left[(n-m+1)(n+m)\right]^2$$

A3.166

$$\begin{Bmatrix} A_{nm} \\ B_{nm} \end{Bmatrix} = \frac{1}{c}\left(2-\delta_m^0\right)(-1)^m(2n+1)\left[\frac{(n-m)!}{(n+m)!}\right]^2 \cdot \overline{D} \cdot \nabla\left[P_n^m(\eta_0)P_n^m(\xi_0)\begin{Bmatrix}\cos m\varphi_0 \\ \sin m\varphi_0\end{Bmatrix}\right]$$

A3.167

Cuffin and Cohen evaluate the source at $\vartheta_0 = 0$, thus there are no B_{nm} multipole components. They also use a left handed coordinate system and thus the i and j components are reversed compared to this thesis.

So for this thesis the A_{nm} $\hat{j}$ components should be the same as Cuffin and Cohen $\hat{i}$ components:

$$\overline{B}_v(\eta_e,\xi)\ \hat{j} = -\frac{\mu}{4\pi}\frac{1}{\left(\eta_e^2-1\right)^{1/2}}\sum_{n=1}^{\infty}\frac{1}{2}\cos\varphi_e \quad \text{multiplied by}$$

A3.168

the Legendre terms contained in:

1] the term containing β^{*}_{nm} evaluated at $m = m-1$

plus

2] the term containing β^{**}_{nm} evaluated at $m = m+1$

- the $A_{n,m-1}\beta^{*}_{n,m-1}$ $\hat{j}$ term =

$$\frac{2}{c}(-1)^{m-1}(2n+1)\left[\frac{(n-m+1)!}{(n+m-1)!}\right]^2 \frac{Q_n^m(\eta_e)P_n^m(\xi_e)P_n^m(\eta_e)}{(n-m+1)(n+m)P_n'^{m-1}(\eta_e)} \cdot \overline{D}\cdot\nabla\left\{P_n^{m-1}(\xi_0)P_n^{m-1}(\eta_0)\right\}$$

A3.169

$$= \frac{2}{c}(-1)^{m-1}(2n+1)\left[\frac{(n-m)!}{(n+m)!}\right]^2 (n-m+1)(n+m)\frac{Q_n^m(\eta_e)P_n^m(\xi_e)P_n^m(\eta_e)}{P_n'^{m-1}(\eta_e)} \cdot \overline{D}\cdot\nabla\left\{P_n^{m-1}(\xi_0)P_n^{m-1}(\eta_0)\right\}$$

A3.170

- the $A_{n,m+1}\beta^{**}_{n,m+1}\ \hat{j}$ term =

$$\frac{2}{c}(-1)^{m+1}(2n+1)\left[\frac{(n-m-1)!}{(n+m+1)!}\right]^2\left[(n+m+1)(n-m)\right]^2\frac{Q_n^m(\eta_e)P_n^m(\xi_e)P_n^m(\eta_e)}{P_n'^{m+1}(\eta_e)}\cdot$$
$$\overline{D}\bullet\nabla\left\{P_n^{m+1}(\xi_0)P_n^{m+1}(\eta_0)\right\} \qquad \text{A3.171}$$

$$= -\frac{2}{c}(-1)^{m}(2n+1)\left[\frac{(n-m)!}{(n+m)!}\right]^2\frac{Q_n^m(\eta_e)P_n^m(\xi_e)P_n^m(\eta_e)}{P_n'^{m+1}(\eta_e)}\cdot \qquad \text{A3.172}$$
$$\overline{D}\bullet\nabla\left\{P_n^{m+1}(\xi_0)P_n^{m+1}(\eta_0)\right\}$$

The above two terms are the same as the Cufin and Cohen i term. The Cuffin and Cohen special term is $A_{n1}\beta^{**}_{n1}\ \hat{j}$

Appendix 4. Data from Performance Tests

The Original True Data

Shallow source:

Dipole co-ordinates	x=0.03	y=0.948	z=0.04
Dipole strengths	D_x=0.1	D_y=0.2	D_z=0.3
Quadrupole co-ordinates	x=0.03	y=0.943	z=0.04
Quadrupole strengths - magnetic	Q_{xx}=0.0004545	Q_{yx}=0.0009091	Q_{zx}=0.0013636
	Q_{xy}=0.0013636	Q_{yy}=0.0013636	Q_{zy}=0.0009091
	Q_{xz}=0.0018181	Q_{yz}=0.0004545	Q_{zz}=-0.0018181
Quadrupole strengths - electric	Q_{xx}=0.0004545	Q_{yx}=0.0011364	Q_{zx}=0.001591
		Q_{yy}=0.0013636	Q_{zy}=0.0006818

Table A4-1

Deep source:

Dipole co-ordinates	x=0.03	y=0.768	z=0.04
Dipole strengths	D_x=0.1	D_y=0.2	D_z=0.3
Quadrupole co-ordinates	x=0.03	y=0.763	z=0.04
Quadrupole strengths - magnetic	Q_{xx}=0.0004545	Q_{yx}=0.0009091	Q_{zx}=0.0013636
	Q_{xy}=0.0013636	Q_{yy}=0.0013636	Q_{zy}=0.0009091
	Q_{xz}=0.0018181	Q_{yz}=0.0004545	Q_{zz}=-0.0018181
Quadrupole strengths - electric	Q_{xx}=0.0004545	Q_{yx}=0.0011364	Q_{zx}=0.001591
		Q_{yy}=0.0013636	Q_{zy}=0.0006818

Table A4-2

Performance Tests

Shallow Source:

Electric potential only							
Start position:	x=	y=	z=				
Dipole	0.02	0.86	0.02				
Quadrupole	0.02	0.85	0.02				
					phi Error Squared		
Finish position:	x=	y=	z=				
Dipole	0.03041	0.93826	0.03968	LM	0.00193		
Quadrupole	0.02994	0.94326	0.03999				
Strengths:							
Dipole	Dx=	0.100024	Dy=	0.200142		Dz=	0.300179
Quadrupole	Qxx=	-0.000246	Qxy=	0.001581		Qxz=	0.001543
			Qyy=	0.002633		Qyz=	0.002163
Start position:	x=	y=	z=				
Dipole	0.02	0.91	0.02				
Quadrupole	0.02	0.85	0.02				
					phi Error Squared		
Finish posititon:	x=	y=	z=				
Dipole	0.02998	0.94799	0.04001		0.00001		
Quadrupole	0.03	0.94301	0.04				
Strengths:							
Dipole	Dx=	0.099999	Dy=	0.200001		Dz=	0.300002
Quadrupole	Qxx=	0.000456	Qxy=	0.001139		Qxz=	0.001593
			Qyy=	0.001366		Qyz=	0.000682
Start position:	x=	y=	z=				
Dipole	0.02	0.86	0.02				
Quadrupole	0.02	0.85	0.02				
					phi Error Squared		
Finish position:	x=	y=	z=				
Dipole	0.02998	0.94799	0.04001	GM	0		
Quadrupole	0.03	0.94301	0.04				
Strengths:							
Dipole	Dx=	0.099999	Dy=	0.200001		Dz=	0.300002
Quadrupole	Qxx=	0.000456	Qxy=	0.001139		Qxz=	0.001593
			Qyy=	0.001366		Qyz=	0.000682

Table A4-3

Shallow Source:

Magnetic Field Bx only						
Rank = 10						
Start position:	x=	y=	z=			
Dipole	0.02	0.9	0.02			
Quadrupole	0.02	0.85	0.02			
				B Error Squared		
Finish position:	x=	y=	z=			
Dipole	0.03022	0.94817	0.04003	416		
Quadrupole	0.02993	0.94322	0.04003			
Strengths:						
Dipole	Dx=	-	Dy=	0.2002	Dz=	0.2997
Quadrupole	Qxx=	0.0032	Qyx=	0.0195	Qzx=	0.0005
	Qxy=	0.0014	Qyy=	-0.002	Qzy=	0.0009
	Qxz=	0.0019	Qyz=	0.0005	Qzz=	-
Rank = 12						
Start position:	x=	y=	z=			
Dipole	0.02	0.9	0.02			
Quadrupole	0.02	0.85	0.02			
				B Error Squared		
Finish position:	x=	y=	z=			
Dipole	0.03	0.948	0.04	1137		
Quadrupole	0.03	0.943	0.04			
Strengths:						
Dipole	Dx=	0.1038	Dy=	0.2	Dz=	0.3
Quadrupole	Qxx=	0.0032	Qyx=	-0.0004	Qzx=	0.0014
	Qxy=	0.0014	Qyy=	0.0033	Qzy=	0.0009
	Qxz=	0.0018	Qyz=	0.0005	Qzz=	0.0011

Table A4-4

Shallow Source:

Magnetic Field By only						
Rank = 10						
Start position:	x=	y=	z=			
Dipole	0.02	0.9	0.02			
Quadrupole	0.02	0.85	0.02			
				B Error Squared		
Finish position:	x=	y=	z=			
Dipole	0.02999	0.93611	0.04004	0.00246521		
Quadrupole	0.03	0.94301	0.04004			
Strengths:						
Dipole	Dx=	0.0997	Dy=	-	Dz=	0.3021
Quadrupole	Qxx=	0.0011	Qyx=	-0.0001	Qzx=	0.0014
	Qxy=	0.0083	Qyy=	-	Qzy=	0.0176
	Qxz=	0.0018	Qyz=	-0.0024	Qzz=	-0.0368
Rank = 12						
Start position:	x=	y=	z=			
Dipole	0.02	0.9	0.02			
Quadrupole	0.02	0.85	0.02			
				B Error Squared		
Finish position:	x=	y=	z=			
Dipole	-0.00242	0.96102	0.03682	3.35E+08		
Quadrupole	0.01623	0.91723	0.03774			
Strengths:						
Dipole	Dx=	0.0851	Dy=	63.3633	Dz=	-0.0625
Quadrupole	Qxx=	-0.0002	Qyx=	0.0017	Qzx=	0.0017
	Qxy=	0.8307	Qyy=	-12.0548	Qzy=	-1.6397
	Qxz=	-0.0008	Qyz=	-0.0258	Qzz=	1.8586

Table A4-5

Shallow Source:

Electric Potential + Magnetic Field Bx						
Rank = 12						
Start position:	x=	y=	z=			
Dipole	0.02	0.9	0.02		phi Error Squared	
Quadrupole	0.02	0.85	0.02		0.15401	
Finish position:	x=	y=	z=		B Error Squared	
Dipole	0.03023	0.93723	0.03987	LM	0.26	
Quadrupole	0.02997	0.94316	0.04002			
Strengths:						
Dipole	Dx=	0.099983	Dy=	0.200411	Dz=	0.300339
Quadrupole	Qxx=	0.0012	Qyx=	0	Qzx=	0.0013
	Qxy=	0.0083	Qyy=	0	Qzy=	0.0133
	Qxz=	0.0019	Qyz=	-0.0022	Qzz=	-0.0401
Rank = 12						
Start position:	x=	y=	z=			
Dipole	0.02	**0.92**	0.02		phi Error Squared	
Quadrupole	0.02	0.85	0.02		0.058	
Finish position:	x=	y=	z=		B Error Squared	
Dipole	0.02999	0.94799	0.04	GM	0.31	
Quadrupole	0.03	0.943	0.04			
Strengths:						
Dipole	Dx=	0.099683	Dy=	0.199999	Dz=	0.300031
Quadrupole	Qxx=	0.0011	Qyx=	0.0009	Qzx=	0.0014
	Qxy=	0.0056	Qyy=	0	Qzy=	0.0092
	Qxz=	0.0018	Qyz=	0.0005	Qzz=	-0.0377

Table A4-6

Deep Source:

Electric Potential only						
Start position:	x=	y=	z=			
Dipole	0.02	0.66	0.02		phi Error Squared	
Quadrupole	0.02	0.65	0.02		0.00066	
Finish position:	x=	y=	z=			
Dipole	0.02873	0.75515	0.04067	LM		
Quadrupole	0.02994	0.76237	0.04006			
Strengths:						
Dipole	Dx=	0.100095	Dy=	0.200033	Dz=	0.300126
Quadrupole	Qxx=	-0.00025	Qxy=	0.001904	Qxz=	0.001754
			Qyy=	0.003105	Qyz=	0.002539
Start position:	x=	y=	z=			
Dipole	0.02	0.66	0.02		phi Error Squared	
Quadrupole	0.02	0.65	0.02		0.00001	
Finish position:	x=	y=	z=			
Dipole	0.02984	0.76791	0.04007	GM		
Quadrupole	0.03	0.76307	0.04			
Strengths:						
Dipole	Dx=	0.099995	Dy=	0.200003	Dz=	0.300006
Quadrupole	Qxx=	0.000466	Qxy=	0.001156	Qxz=	0.001611
			Qyy=	0.001376	Qyz=	0.000687

Table A4-7

Deep Source:

Magnetic Field Bx only						
Rank = 12						
Start position:	x=	y=	z=			
Dipole	0.02	0.7	0.02	B Error Squared		
Quadrupole	0.02	0.65	0.02	353		
Finish position:	x=	y=	z=			
Dipole	0.02353	0.76236	0.03853			
Quadrupole	0.04261	0.75152	0.05945			
Strengths:						
Dipole	Dx=	0.0078	Dy=	0.1877	Dz=	0.2874
Quadrupole	Qxx=	0.0021	Qyx=	0.0334	Qzx=	-0.0055
	Qxy=	-0.0001	Qyy=	0.0369	Qzy=	-0.0004
	Qxz=	0.001	Qyz=	-0.002	Qzz=	0.0213

Table A4-8

Deep Source:

Electric Potential and Magnetic Field Bx						
Rank = 12						
Start position:	x=	y=	z=			
Dipole	0.02	0.7	0.02	phi Error Squared		
Quadrupole	0.02	0.65	0.02	0.00014		
Finish position:	x=	y=	z=			
Dipole	0.03009	0.7676	0.04049	B Error Squared		
Quadrupole	0.03012	0.76294	0.04004	5.4		
Strengths:						
Dipole	Dx=	0.0999	Dy=	0.1999	Dz=	0.2999
Quadrupole	Qxx=	0.003	Qyx=	0.0005	Qzx=	0.0012
	Qxy=	0.0014	Qyy=	0.0038	Qzy=	0.001
	Qxz=	0.0018	Qyz=	0.0003	Qzz=	0.0013

Table A4-9

Deep Source:

Electric Potential and Magnetic Fields Bx and By						
Rank = 12						
Start position:	x=	y=	z=			
Dipole	0.02	0.7	0.02	phi Error Squared		
Quadrupole	0.02	0.65	0.02	0.0315		
Finish position:	x=	y=	z=			
Dipole	0.03002	0.76796	0.04114	B Error Squared		
Quadrupole	0.03019	0.76334	0.03989	0.00119465		
Strengths:						
Dipole	Dx=	0.1	Dy=	0.2007	Dz=	0.2998
Quadrupole	Qxx=	0.0003	Qyx=	0.0009	Qzx=	0.0015
	Qxy=	0.0013	Qyy=	0.0013	Qzy=	0.0011
	Qxz=	0.0018	Qyz=	0.0004	Qzz=	-0.0016

Table A4-10

Appendix 5. Dipole Movement in the Presence of an Unknown Quadrupole Source

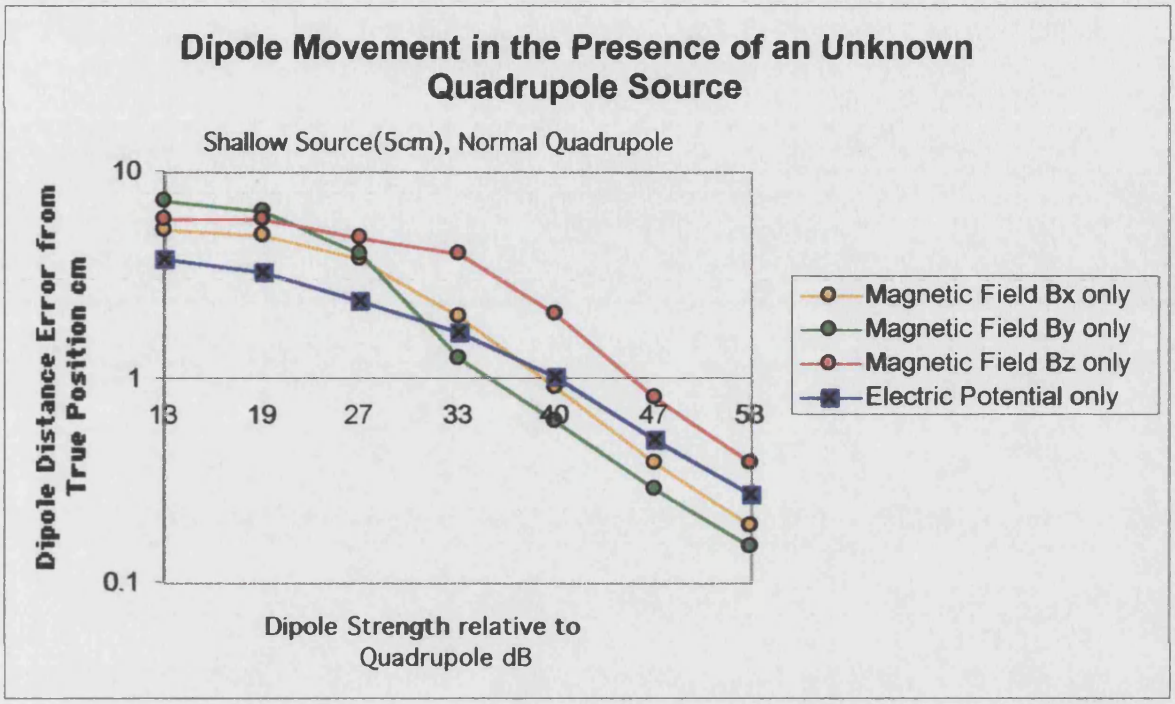


Fig A5-1

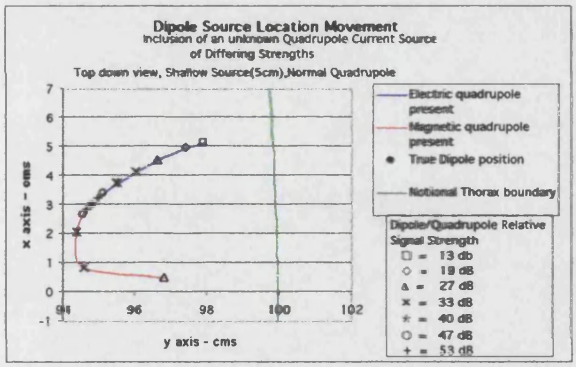


Fig A5-2

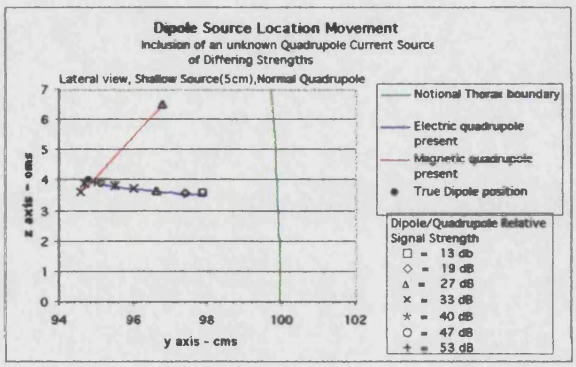


Fig A5-3

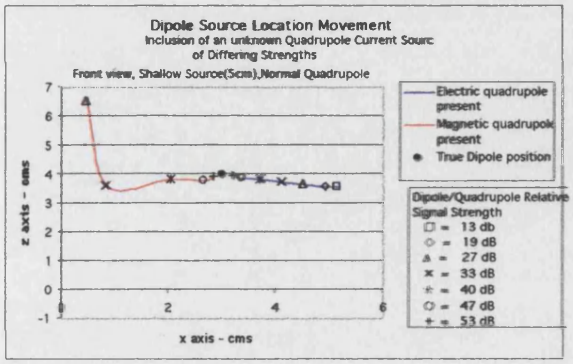


Fig A5-4

Figs A5-1 to A5-4
The dipole inverse models attempting to find the location of a dipole when a quadrupole source of various strengths is included in the same location.
The location is (0.03,0.948.0.04) about 5.1cms below the notional thorax surface.

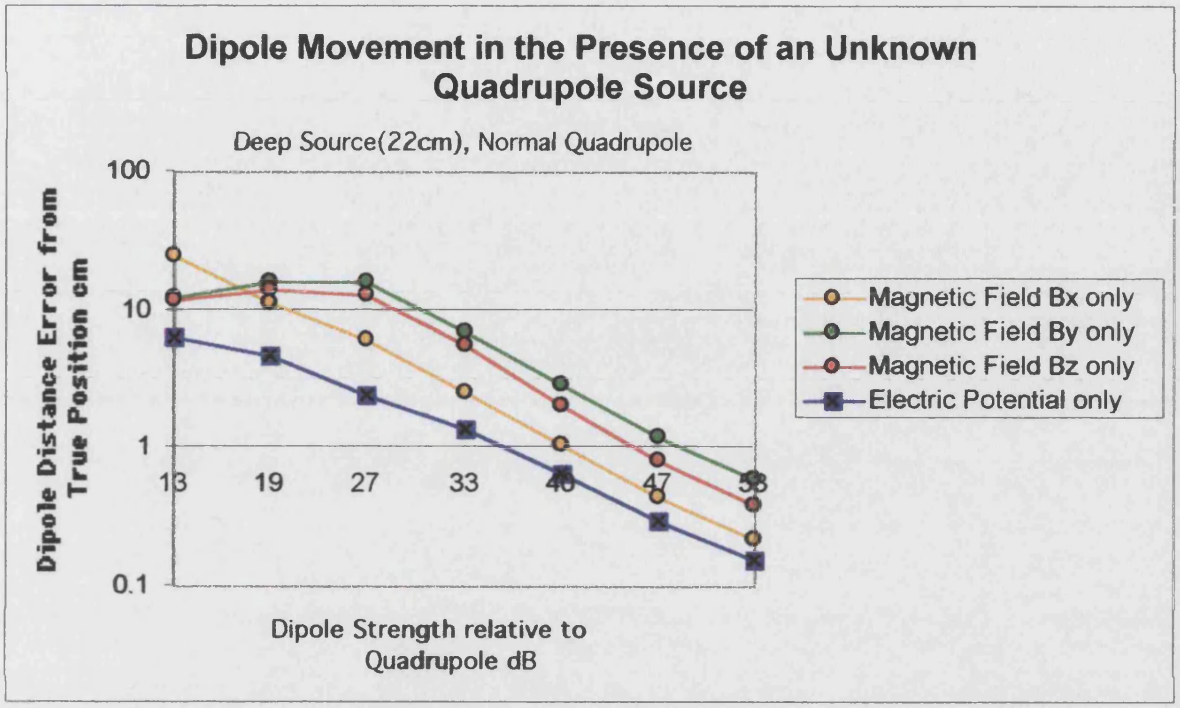


Fig A5-5

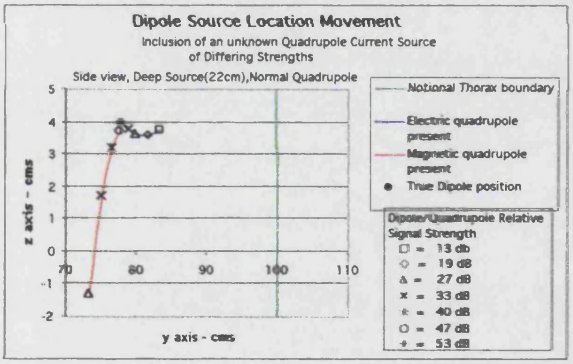


Fig A5-6

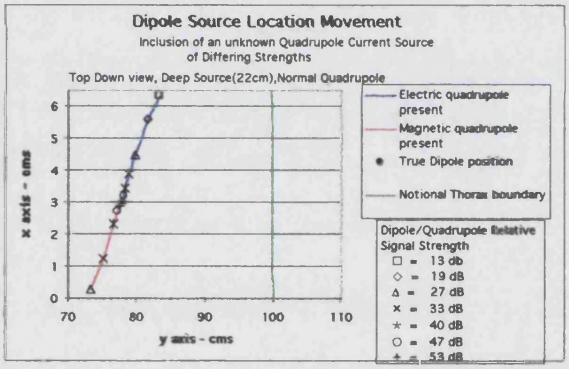


Fig A5-7

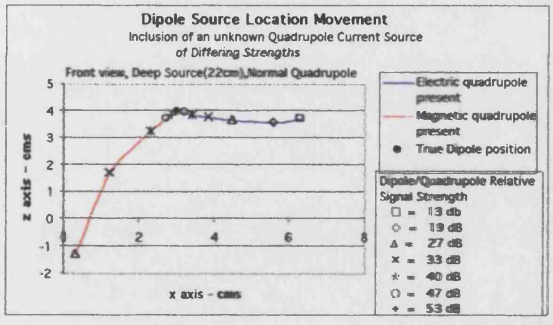


Fig A5-8

Figs A5-5 to A5-8
The dipole inverse models attempting to find the location of a dipole when a quadrupole source of various strengths is included in the same location.
The location is (0.03,0.768.0.04) about 23cms below the notional thorax surface.

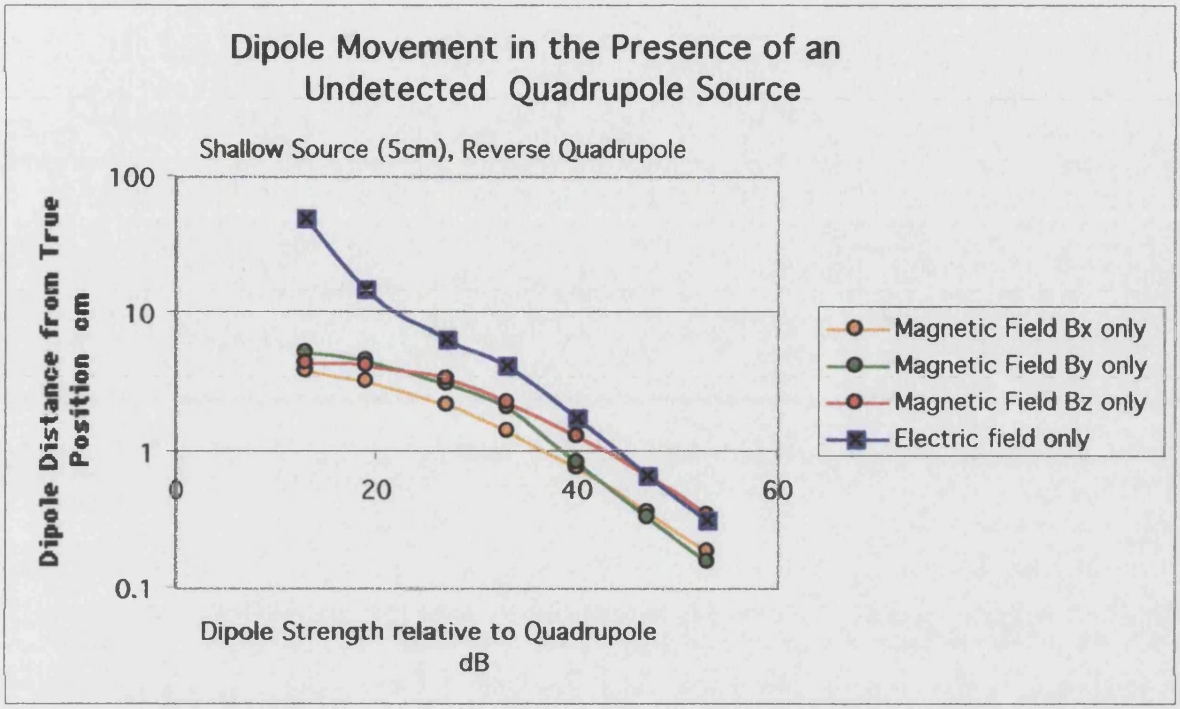


Fig A5-9

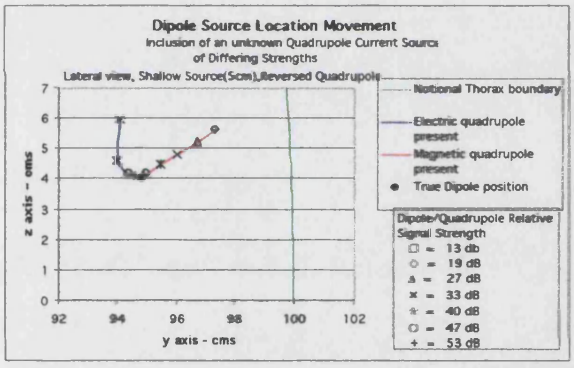


Fig A5-10

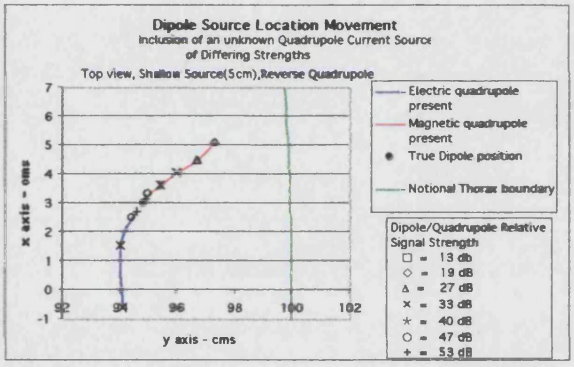


Fig A5-11

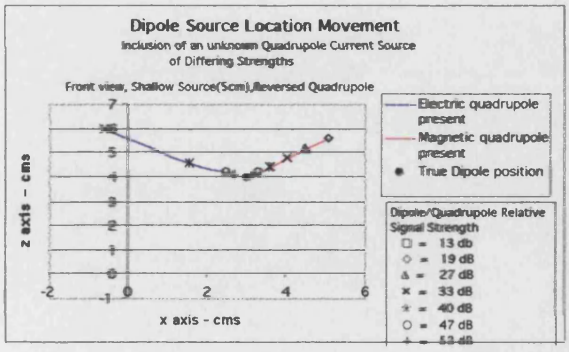


Fig A5-12

Figs A5-10 to A5-12
The dipole inverse models attempting to find the location of a dipole when a quadrupole source of various strengths is included in the same location.
The location is (0.03,0.948.0.04) about 5.1cms below the notional thorax surface. Reversed quadrupole strengths

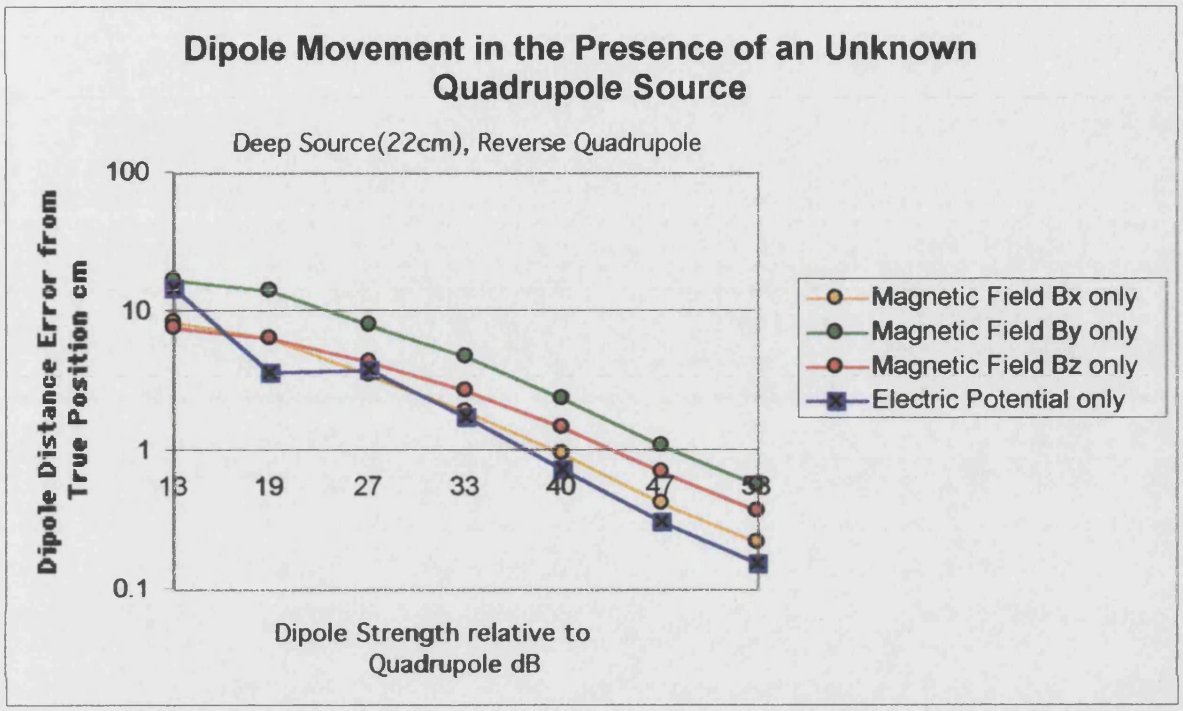


Fig A5-13

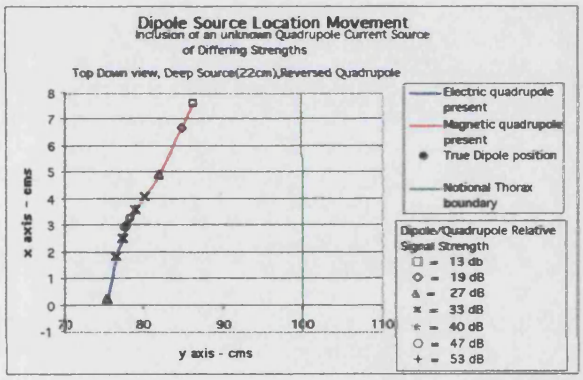


Fig A5-14

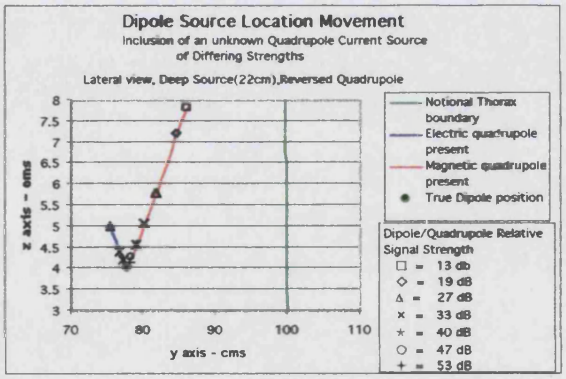


Fig A5-15

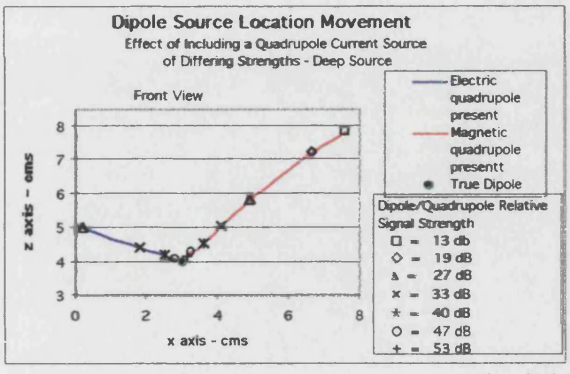


Fig A5-16

Figs A5-13 to A5-16
The dipole inverse models attempting to find the location of a dipole when a quadrupole source of various strengths is included in the same location.
The location is (0.03,0.768.0.04) about 23cms below the notional thorax surface. Reversed quadrupole strengths

Appendix 6. Effect of Noise on the Location of a Dipole and Quadrupole

Electric Potential Inverse

Appendix 6a

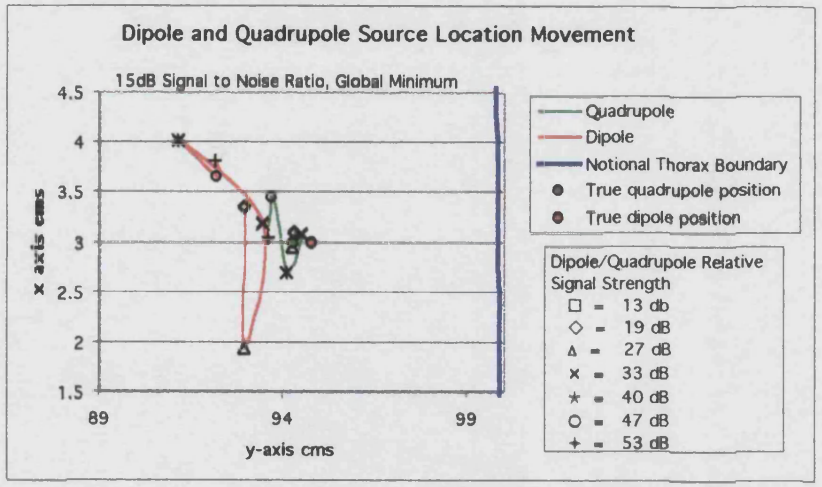


Fig 6a-1

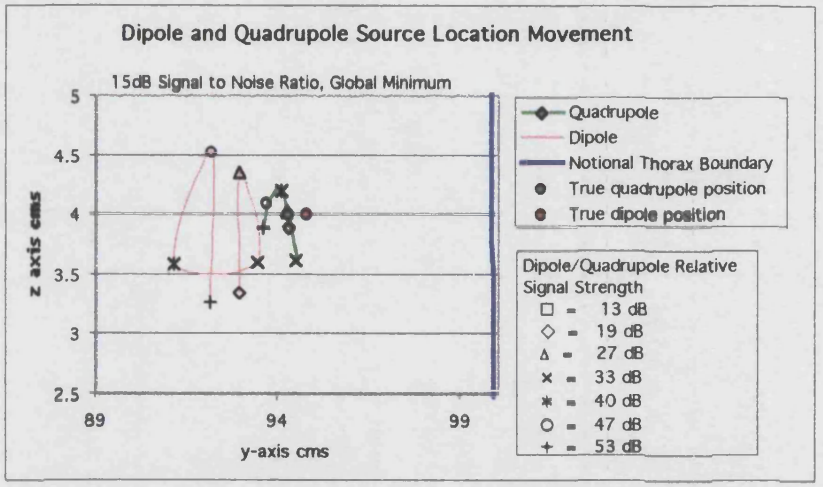


Fig 6a-2

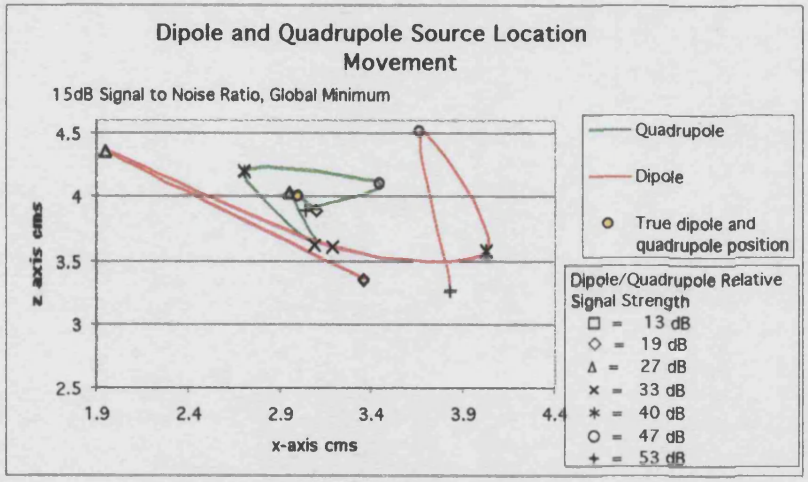


Fig 6a-3

Figs 6a-1 to 6a-3
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using surface electric potentials.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Mean location from 40 to 50 data sets.
Global minimum.

26/7/99

Electric Potential Inverse

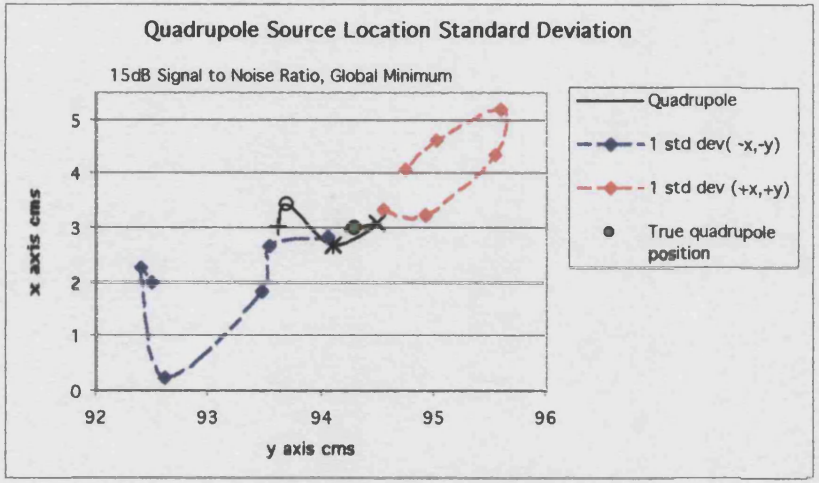


Fig 6a-4

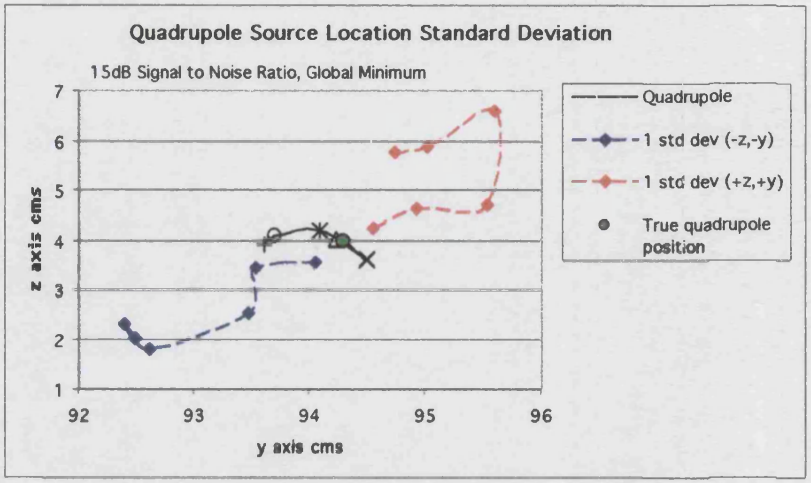


Fig 6a-5

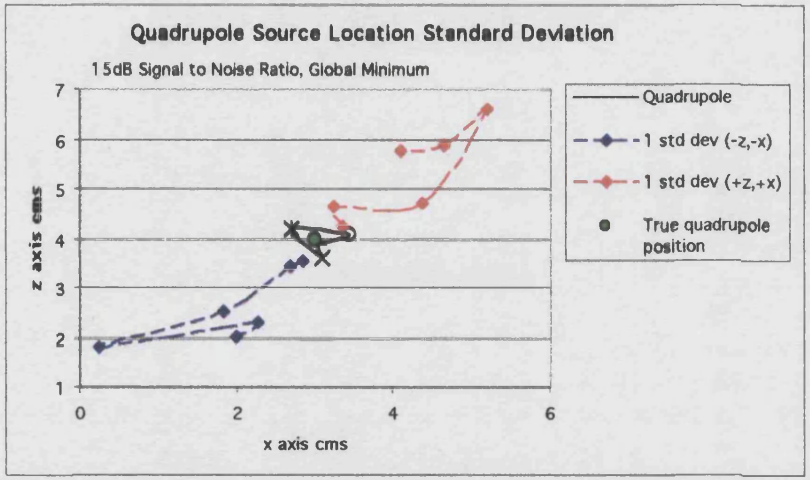


Fig 6a-6

Figs 6a-4 to 6a-6
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using surface electric potentials. Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Standard deviation from 40 to 50 data sets.
Global minimum.

Electric Potential Inverse

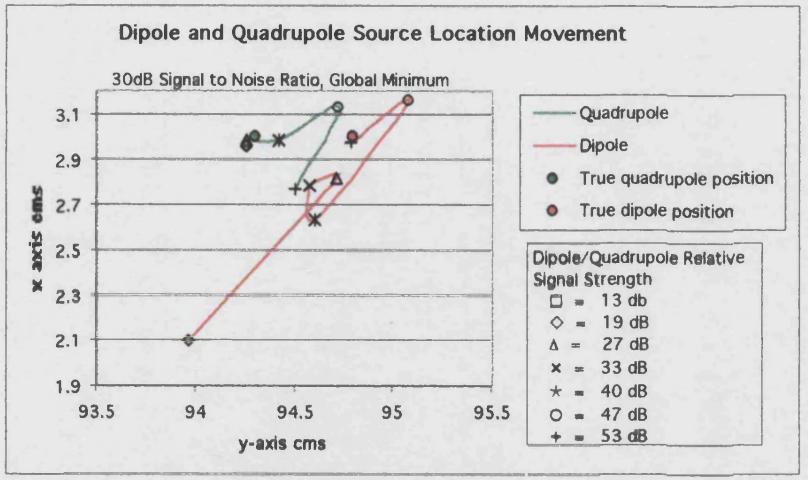


Fig 6a-7

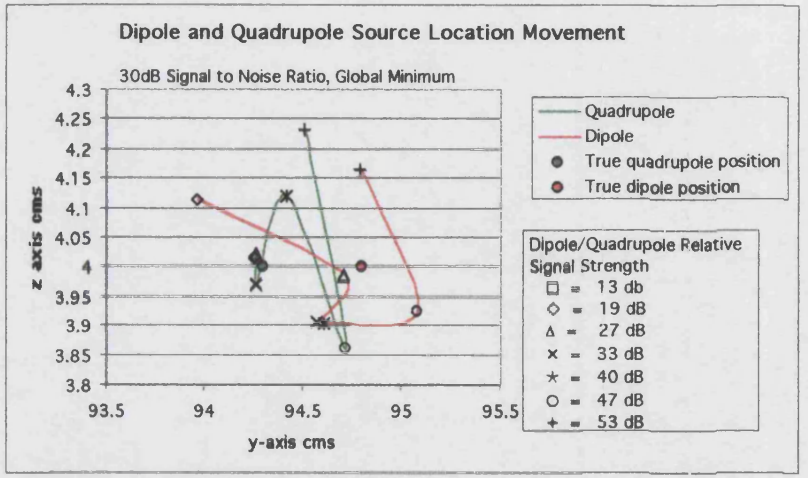


Fig 6a-8

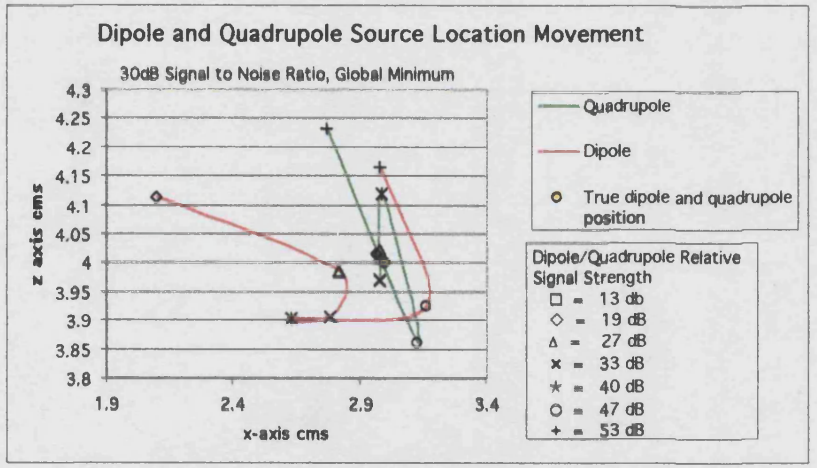


Fig 6a-9

Figs 6a-7 to 6a-9
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using surface electric potentials.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Mean location from 40 to 50 data sets.
Global minimum.

Electric Potential Inverse

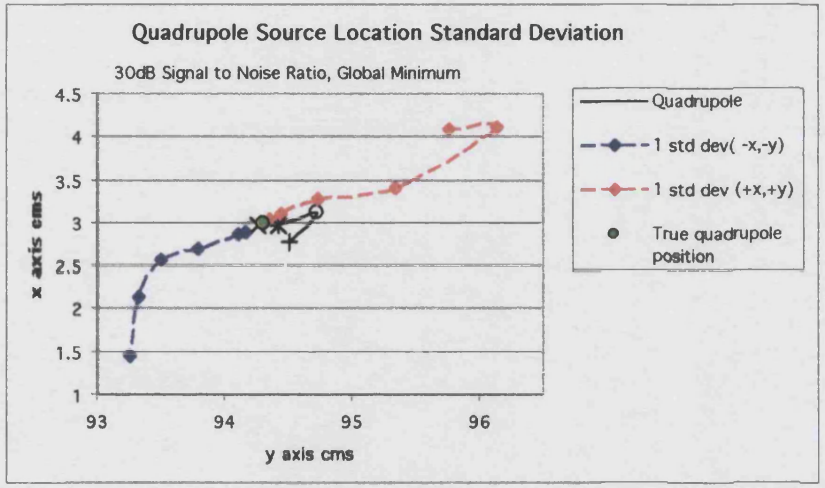


Fig 6a-10

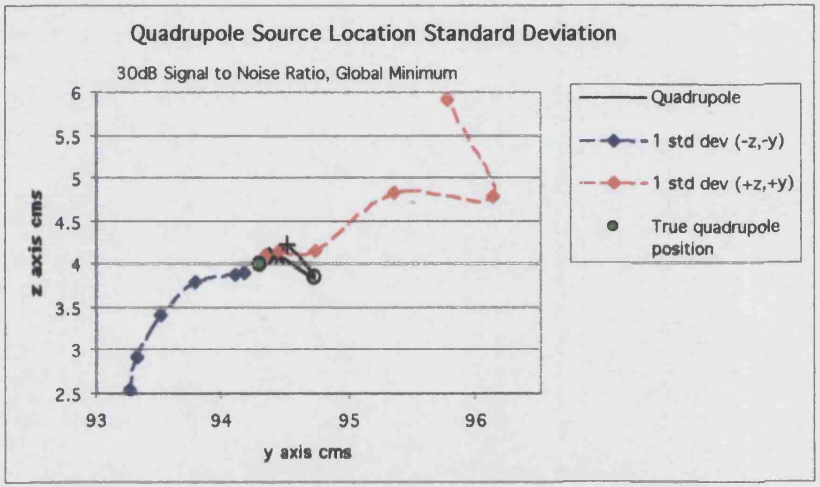


Fig 6a-11

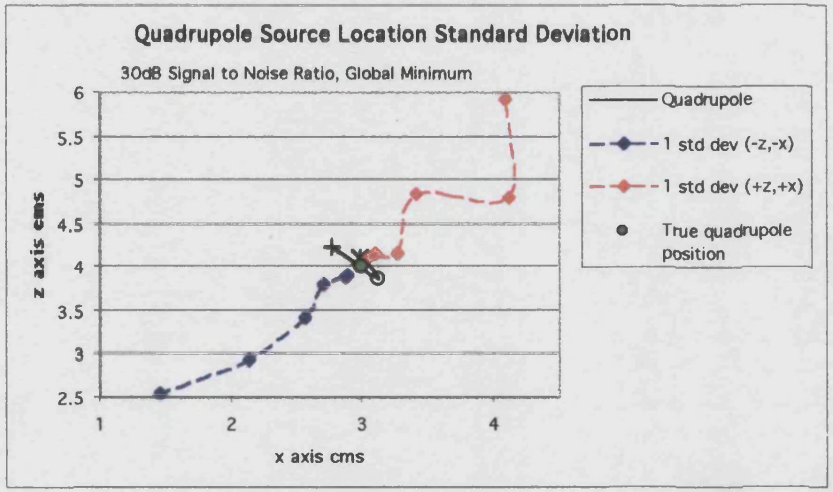


Fig 6a-12

Figs 6a-10 to 6a-12
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using surface electric potentials. Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.
Global minimum.

Electric Potential Inverse

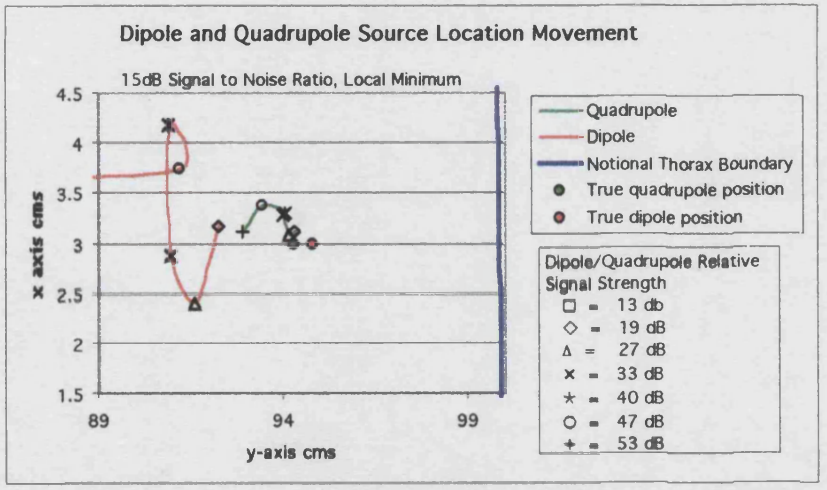


Fig 6a-13

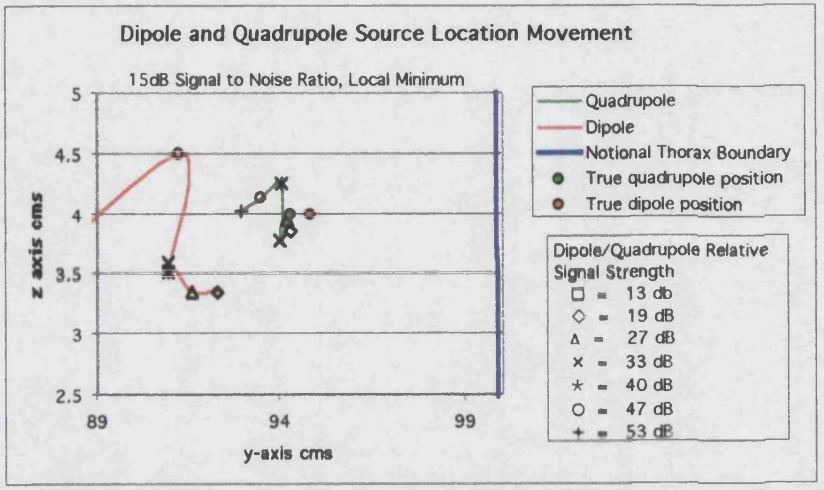


Fig 6a-14

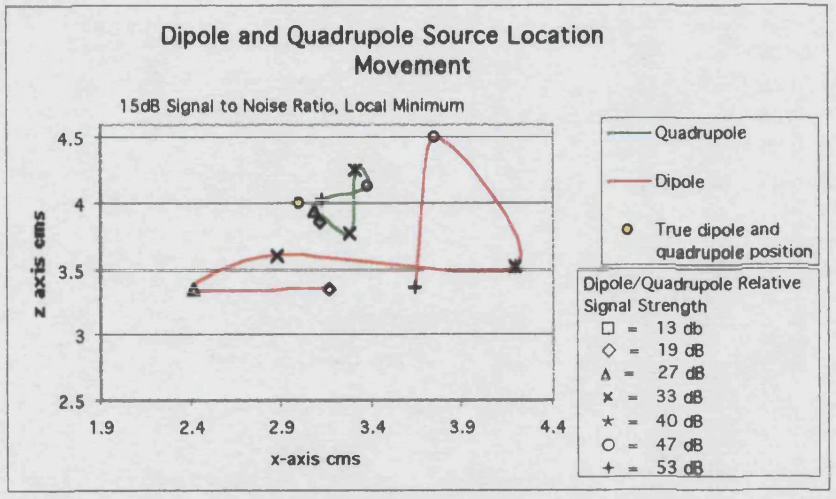


Fig 6a-15

Figs 6a-13 to 6a-15
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using surface electric potentials.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Mean location from 40 to 50 data sets.
Local minimum.

Electric Potential Inverse

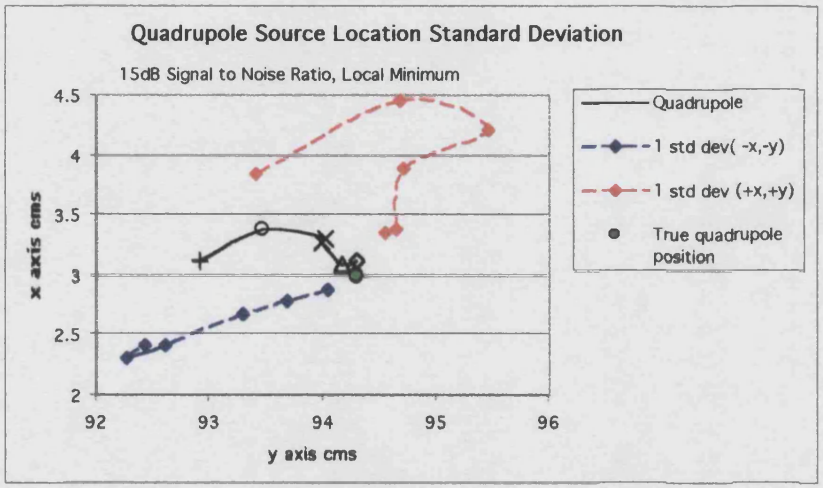


Fig 6a-16

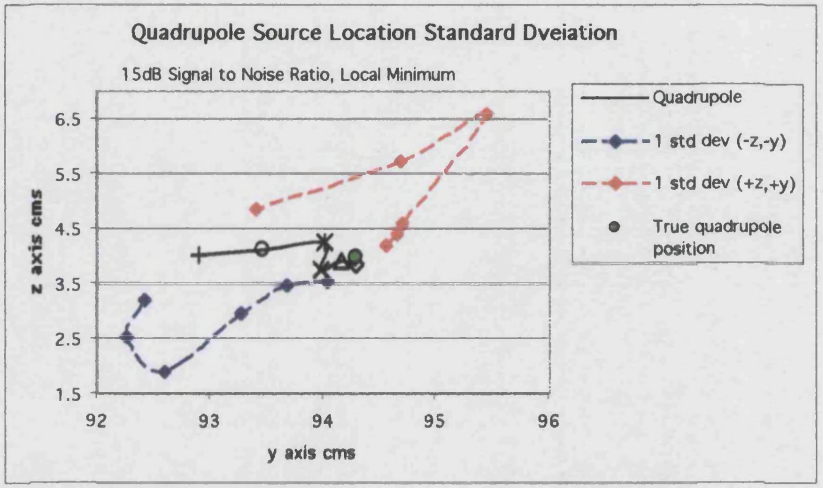


Fig 6a-17

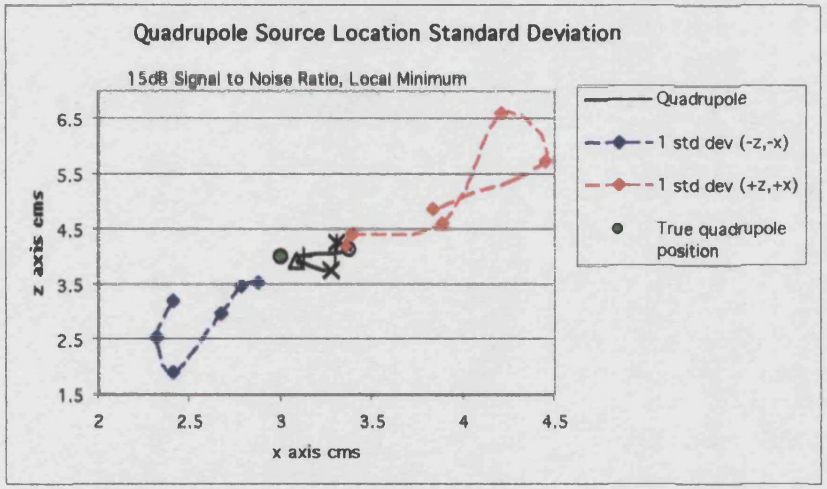


Fig 6a-18

Figs 6a-16 to 6a-18
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using surface electric potentials. Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Standard deviation from 40 to 50 data sets.
Local minimum.

Electric Potential Inverse

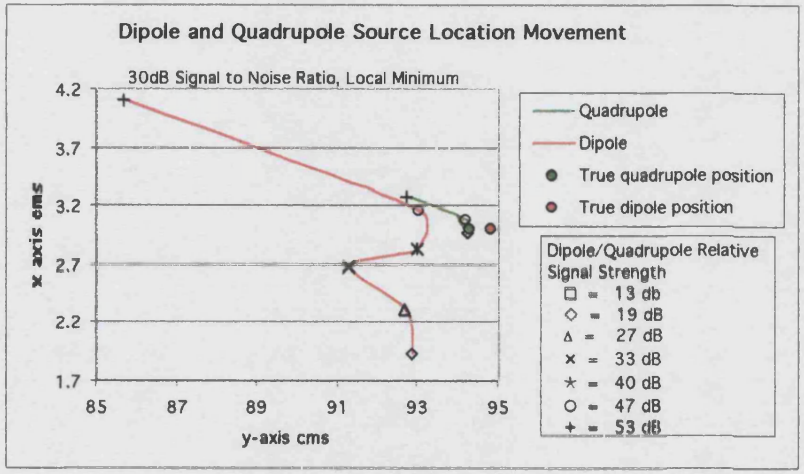


Fig 6a-19

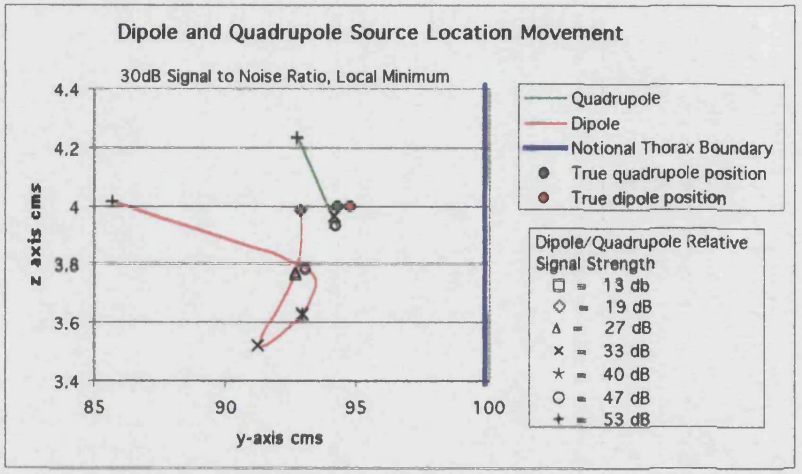


Fig 6a-20

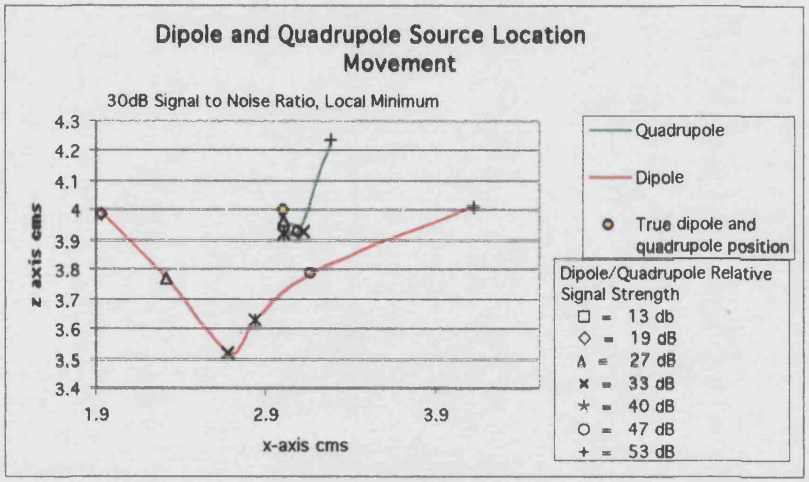


Fig 6a-21

Figs 6a-19 to 6a-21
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using surface electric potentials.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Mean location from 40 to 50 data sets.
Local minimum.

Electric Potential Inverse

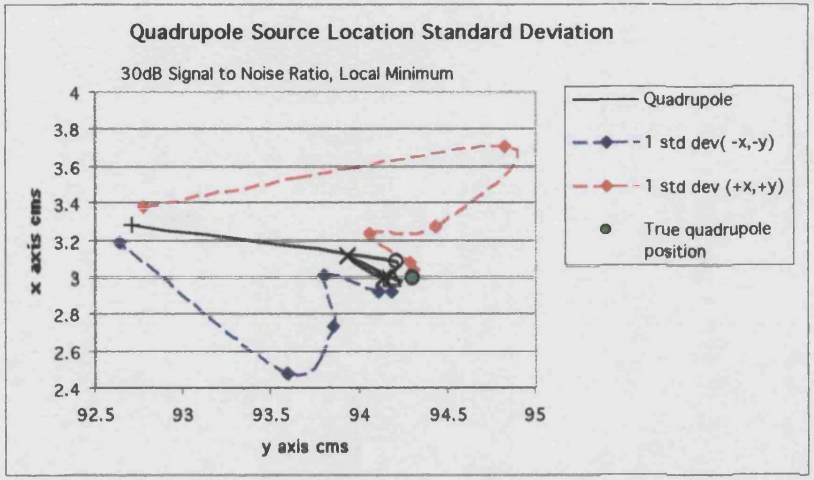


Fig 6a-22

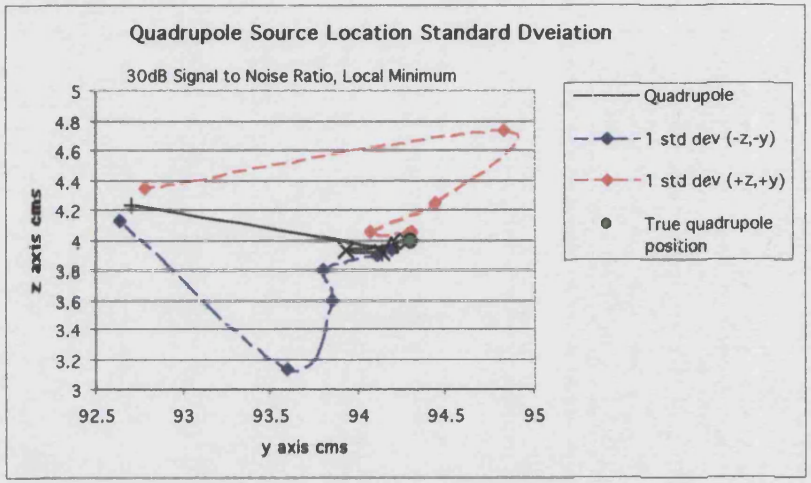


Fig 6a-23

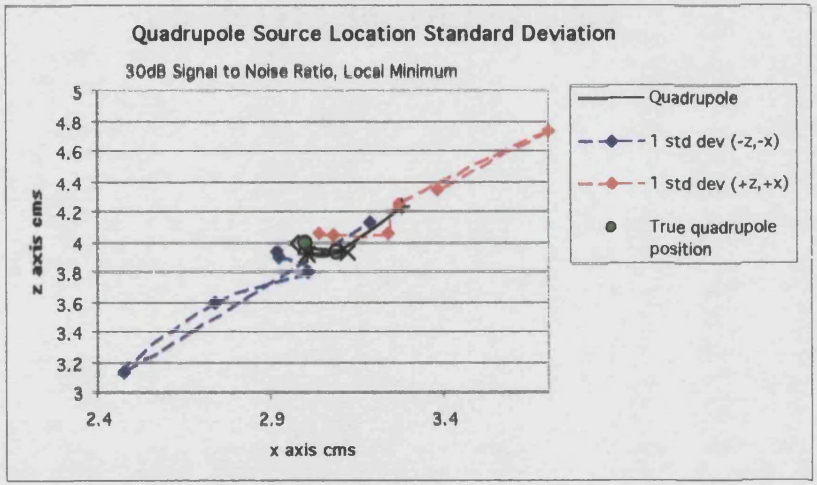


Fig 6a-24

Figs 6a-22 to 6a-24
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using surface electric potentials. Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.
Local minimum.

Magnetic Field B_x Inverse

Appendix 6b

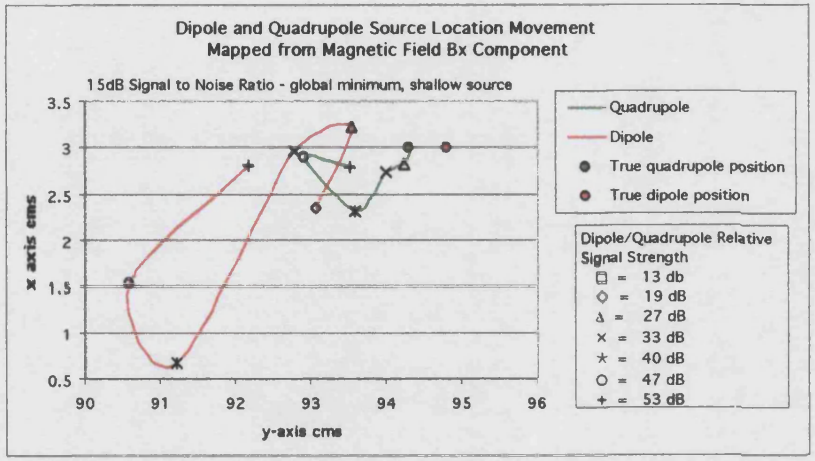


Fig 6b-1

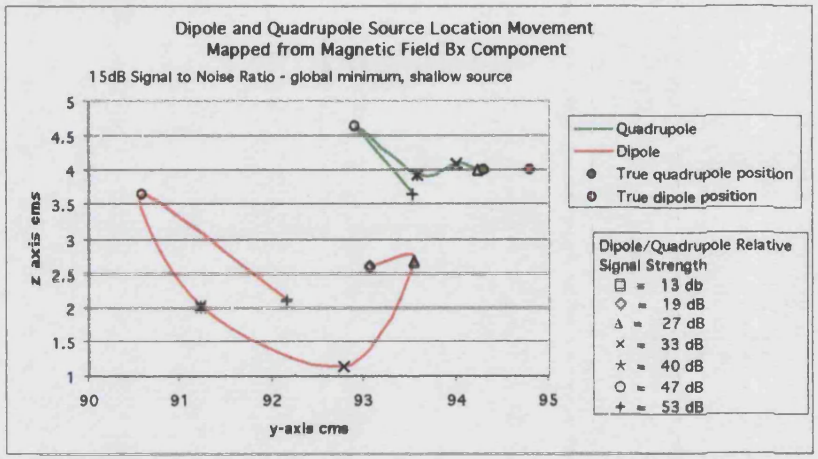


Fig 6b-2

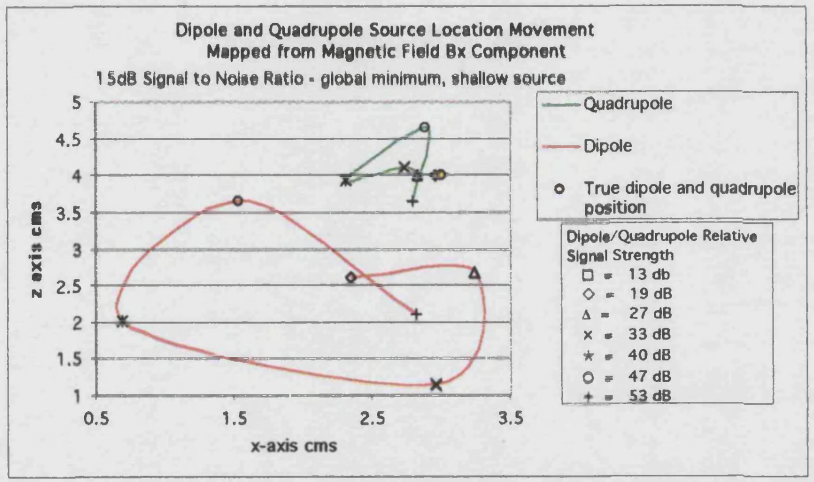


Fig 6b-3

Figs 6b-1 to 6b-3
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Mean location from 40 to 50 data sets.
Global minimum.

Magnetic Field B_x Inverse

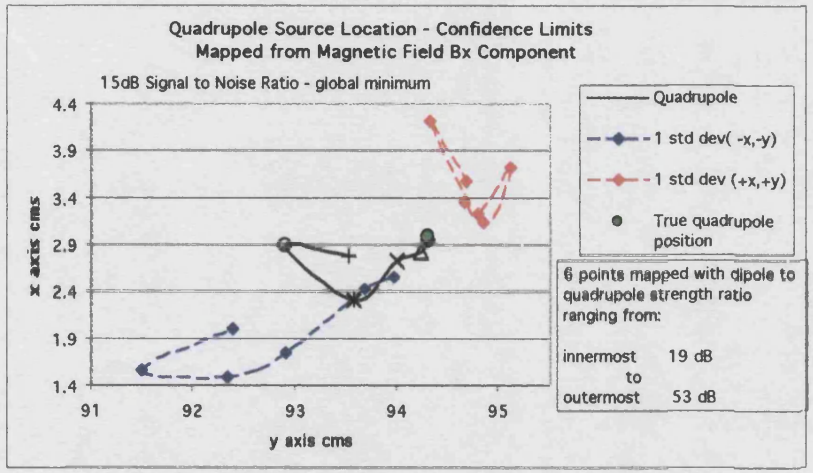


Fig 6b-4

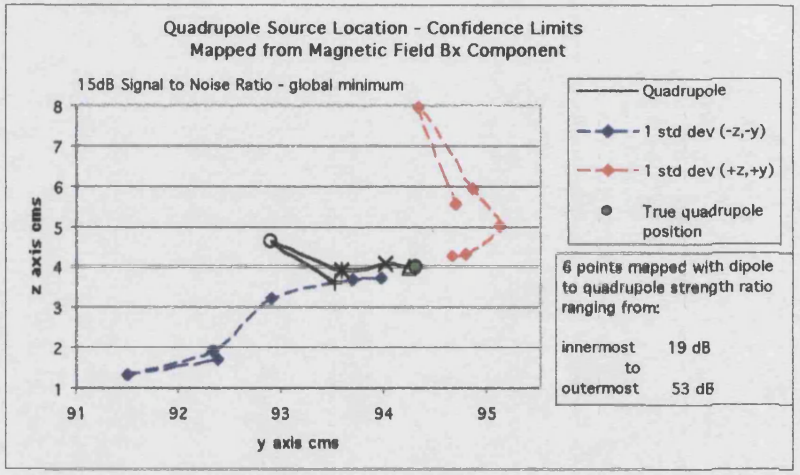


Fig 6b-5

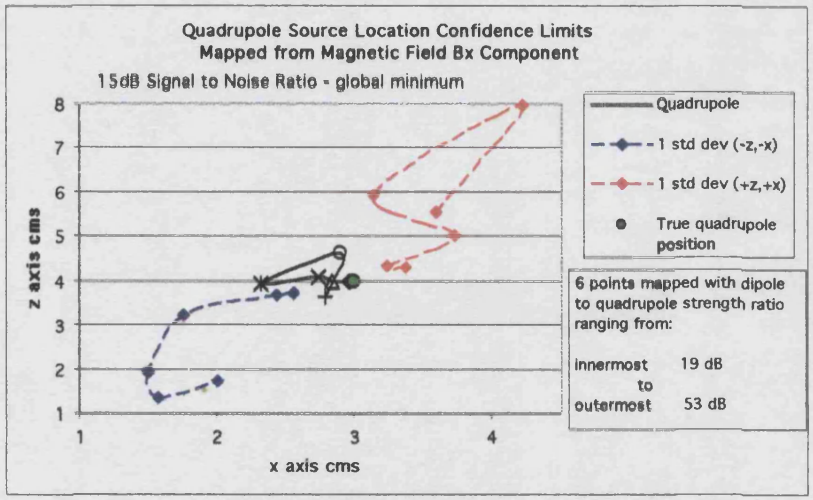


Fig 6b-6

Figs 6b-4 to 6b-6
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface magnetic field B_x. Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Standard deviation from 40 to 50 data sets.
Global minimum.

Magnetic Field B_x Inverse

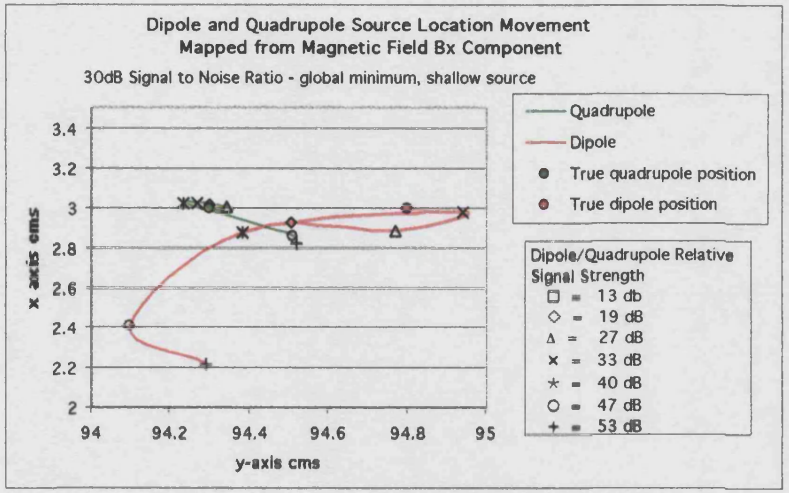


Fig 6b-7

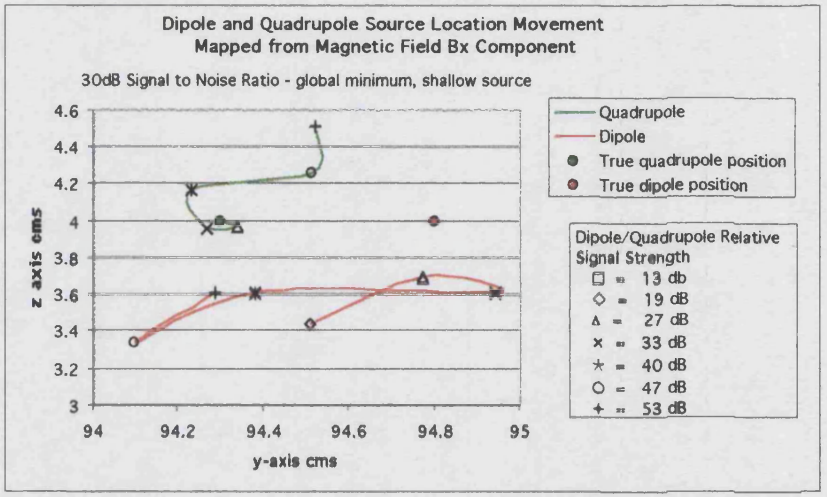


Fig 6b-8

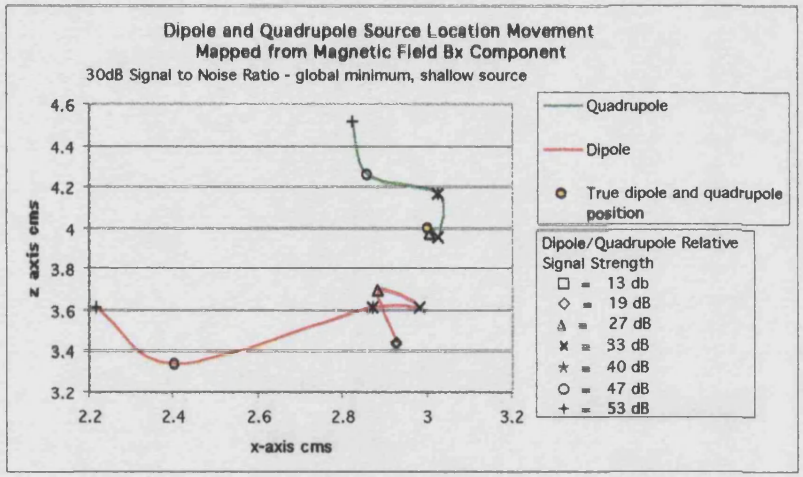


Fig 6b-9

Figs 6b-7 to 6b-9
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Mean location from 40 to 50 data sets.
Global minimum.

Magnetic Field B_x Inverse

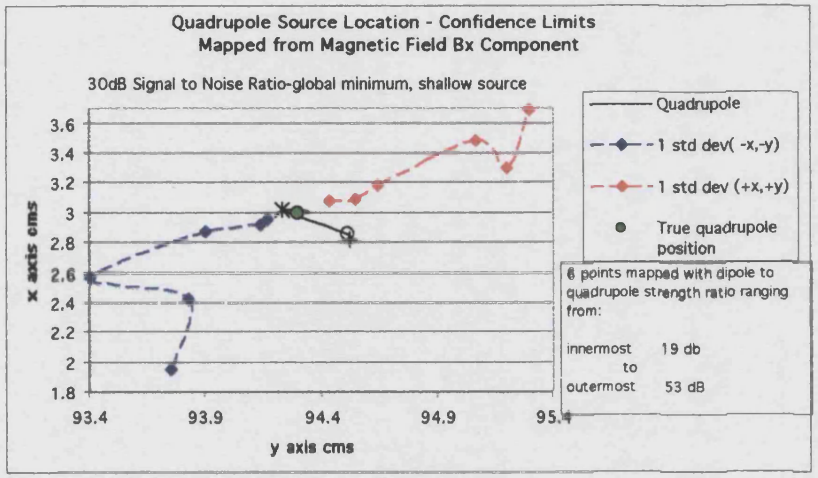


Fig 6b-10

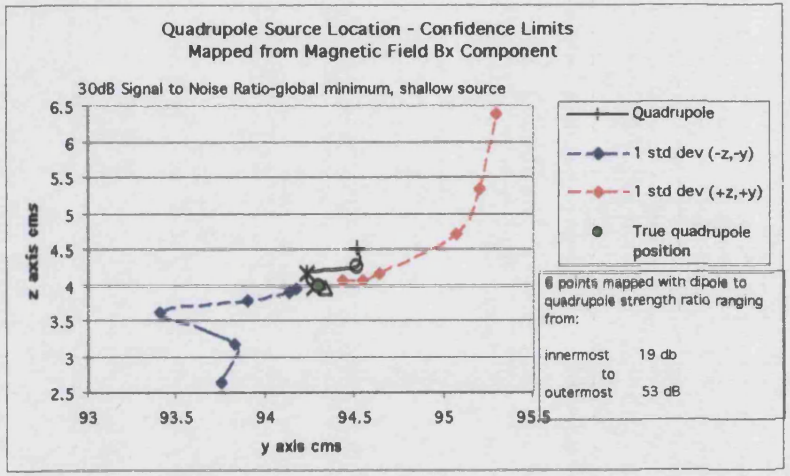


Fig 6b-11

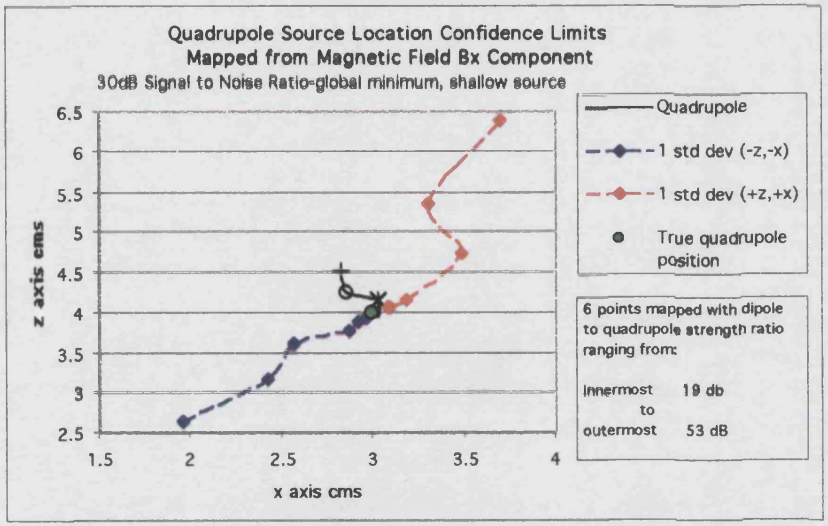


Fig 6b-12

Figs 6b-10 to 6b-12
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface magnetic field B_x. Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.
Global minimum.

Magnetic Field B_x Inverse

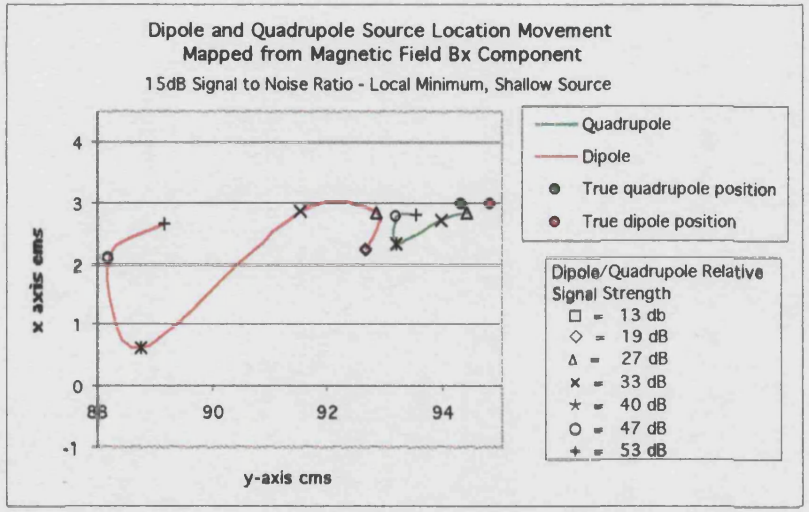


Fig 6b-13

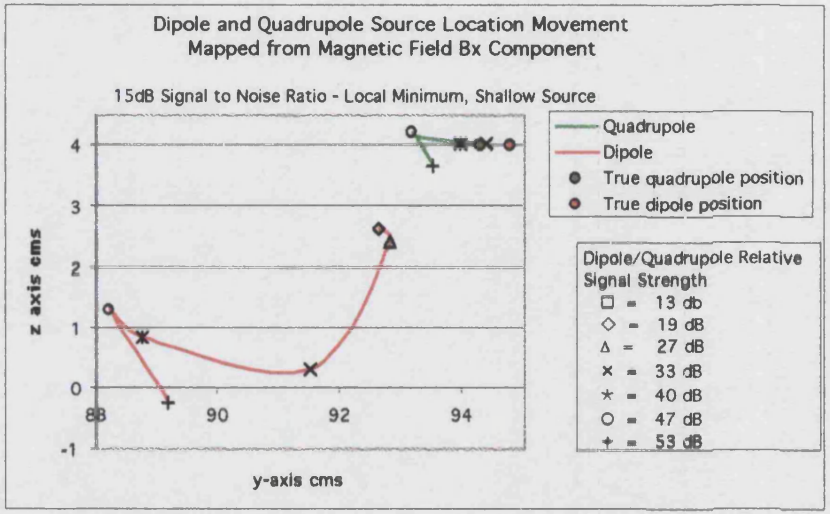


Fig 6b-14

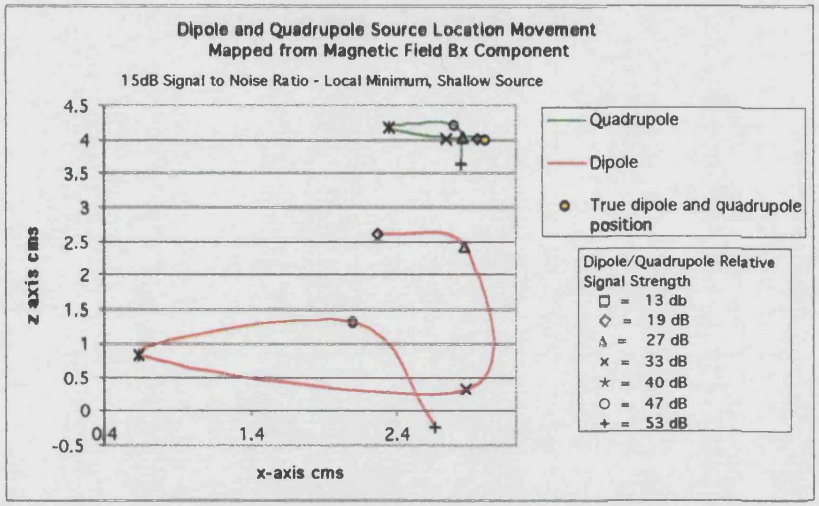


Fig 6b-15

Figs 6b-13 to 6b-15
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Mean location from 40 to 50 data sets.
Local minimum.

26/7/99

Magnetic Field B_x Inverse

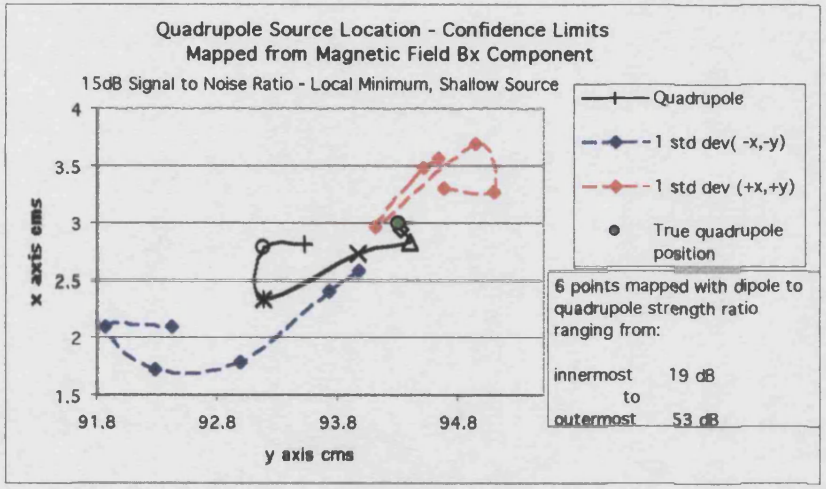


Fig 6b-16

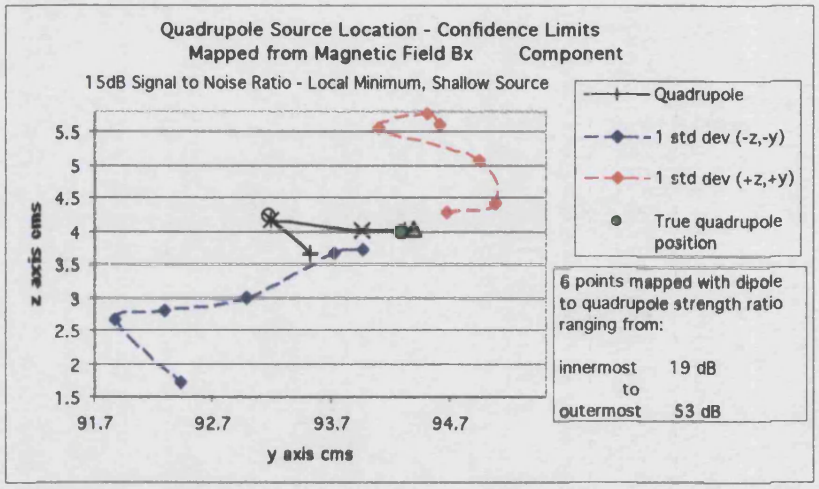


Fig 6b-17

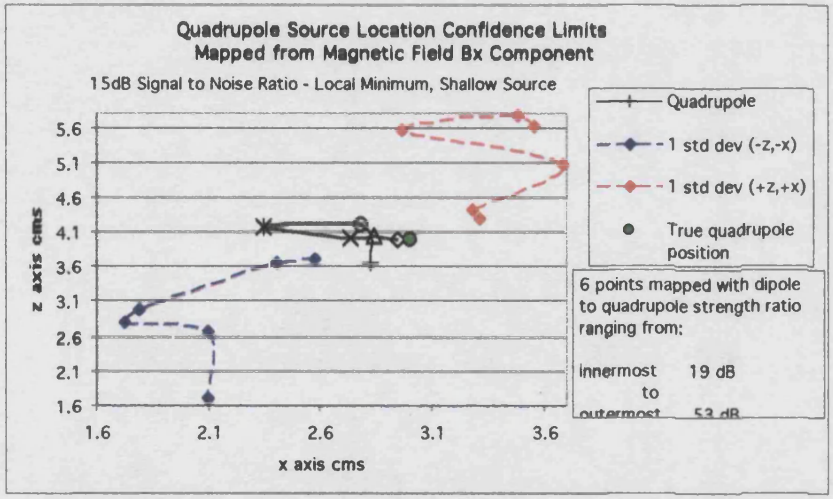


Fig 6b-18

Figs 6b-16 to 6b-18
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface magnetic field B_x. Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Standard deviation from 40 to 50 data sets.
Local minimum.

Magnetic Field B_x Inverse

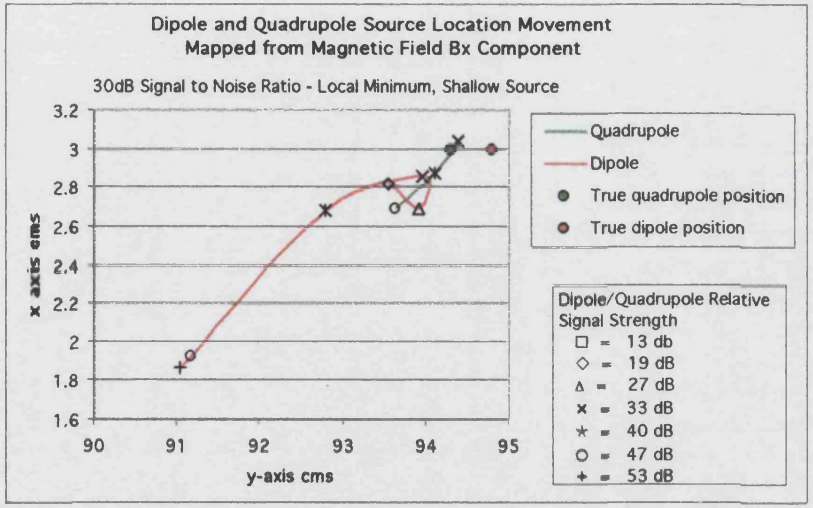


Fig 6b-19

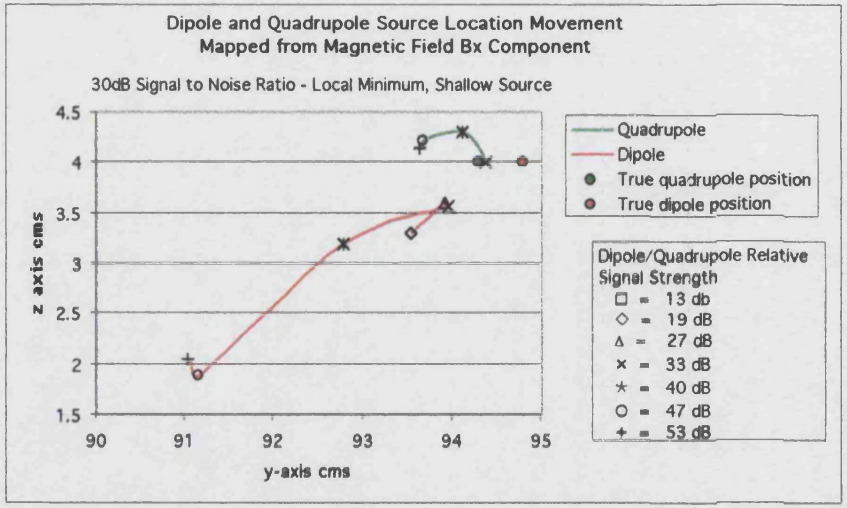


Fig 6b-20

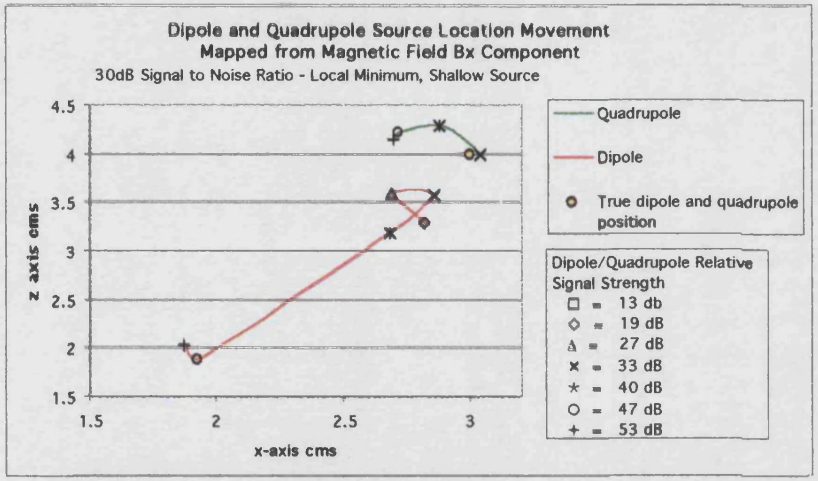


Fig 6b-21

Figs 6b-19 to 6b-21
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Mean location from 40 to 50 data sets.
Local minimum.

Magnetic Field B_x Inverse

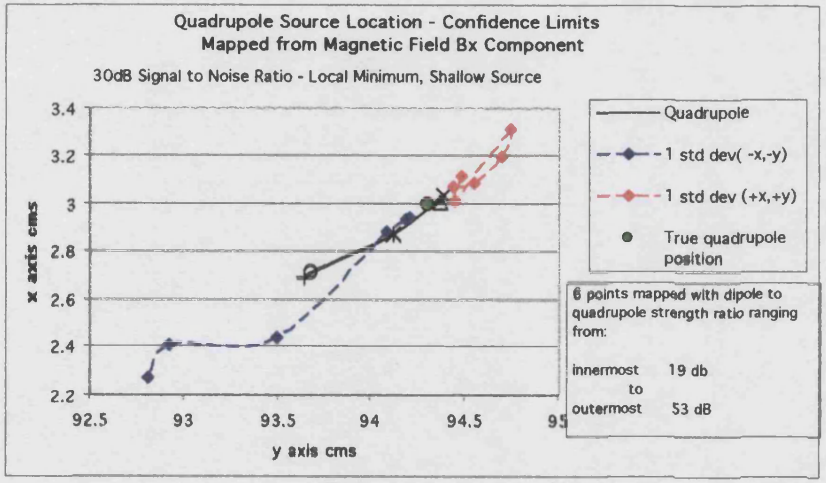


Fig 6b-22

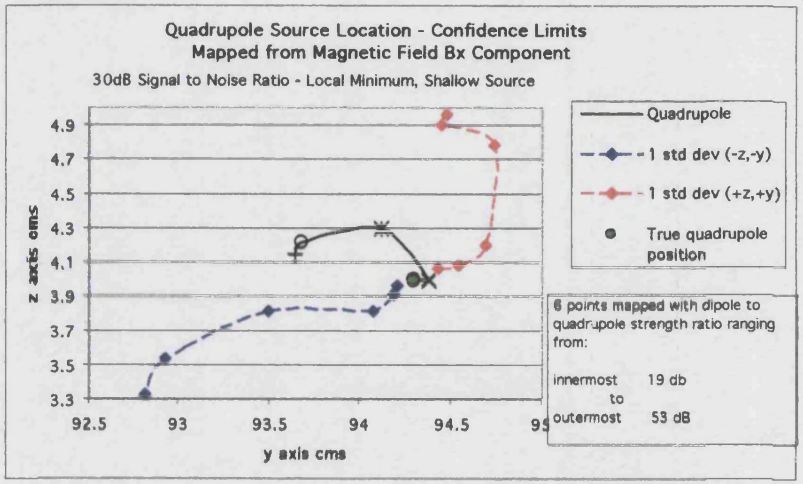


Fig 6b-23

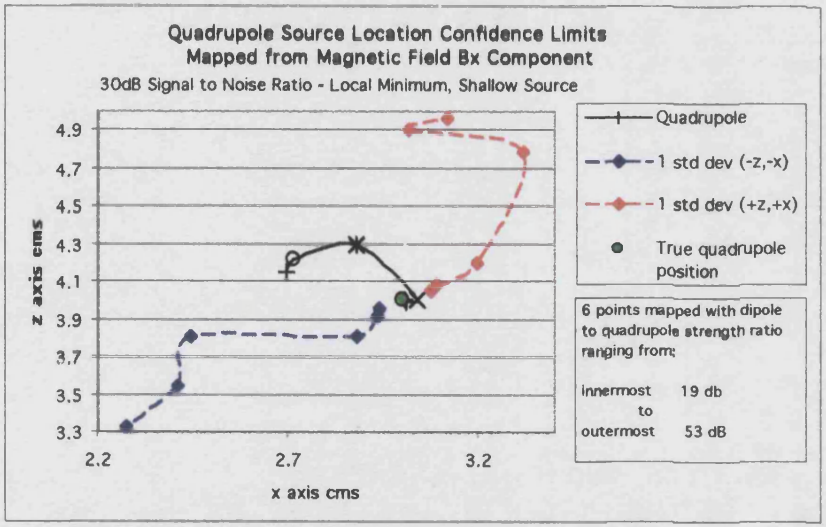


Fig 6b-24

Figs 6b-22 to 6b-24
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface magnetic field B_x. Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.
Local minimum.

26/7/99

Electric Potential andMagnetic Field B_x Inverse

Appendix 6c

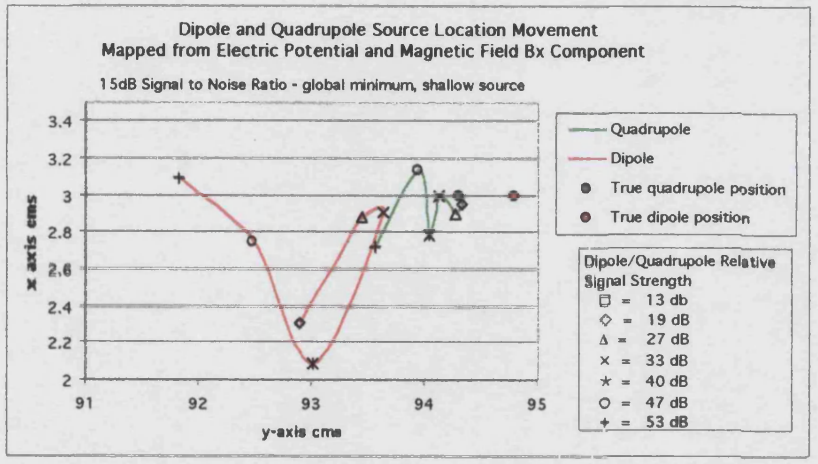


Fig 6c-1

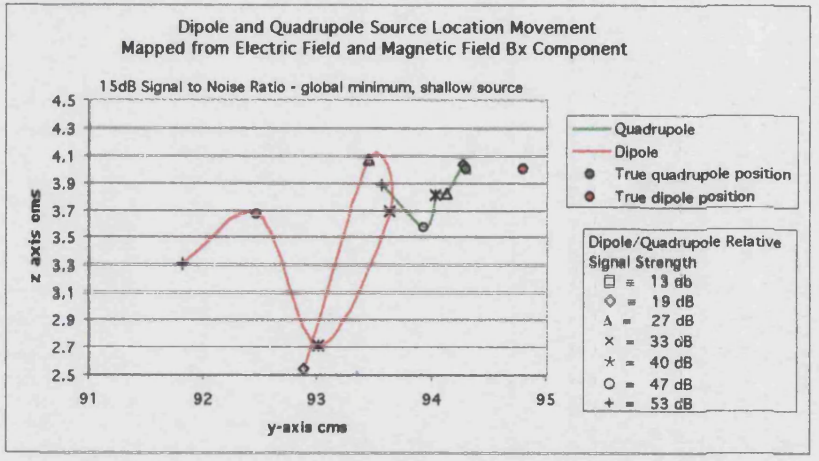


Fig 6c-2

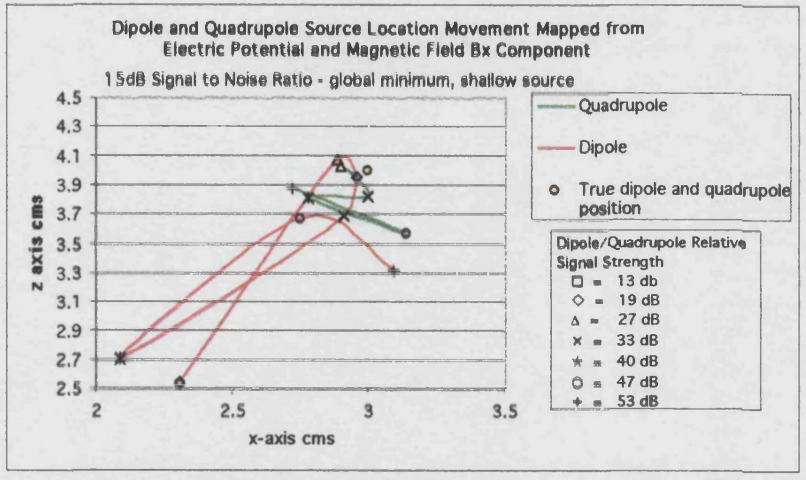


Fig 6c-3

Figs 6c-1 to 6c-3
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface electric potential and magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Mean location from 40 to 50 data sets.
Global minimum.

Electric Potential andMagnetic Field B_x Inverse

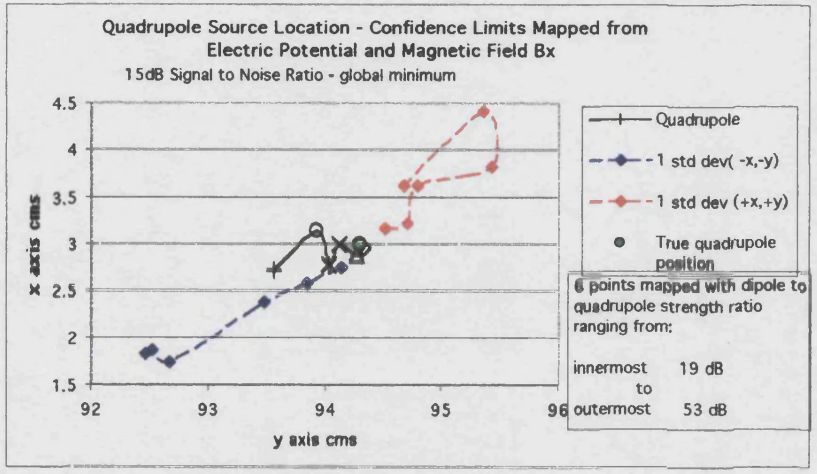


Fig 6c-4

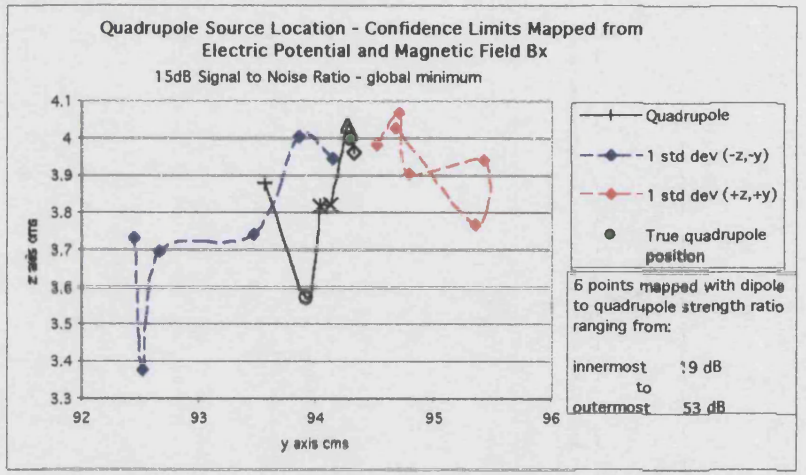


Fig 6c-5

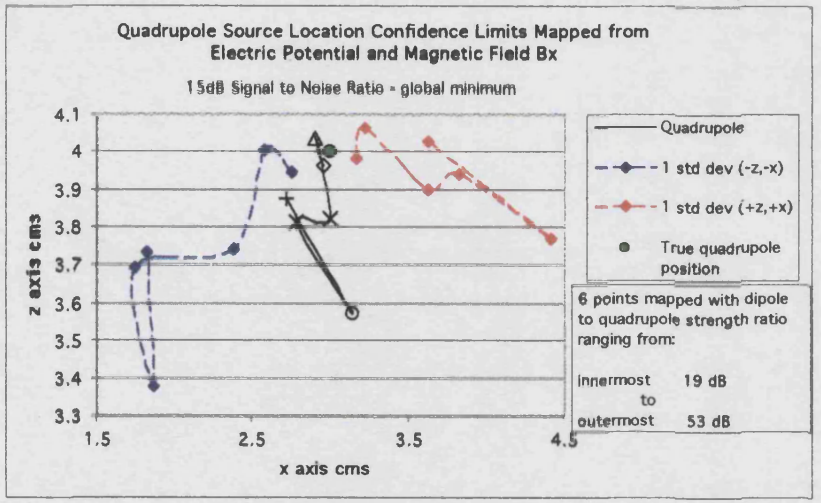


Fig 6c-6

Figs 6c-4 to 6c-6
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface electric potential and magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Standard deviation from 40 to 50 data sets.
Global minimum.

Electric Potential andMagnetic Field B_x Inverse

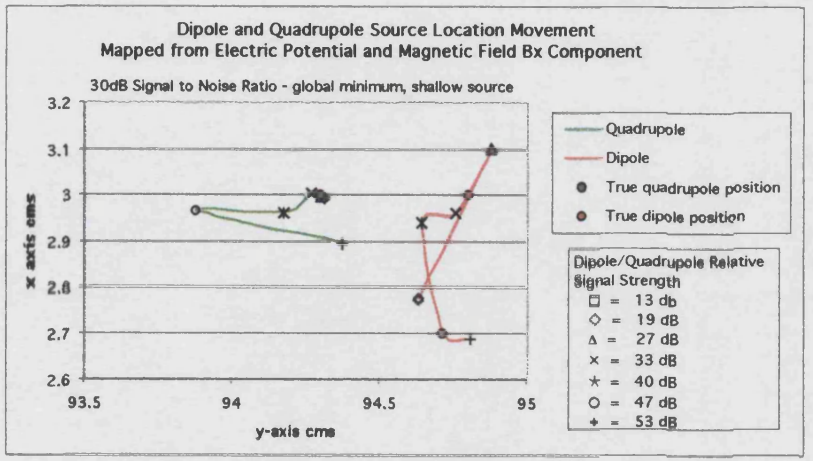


Fig 6c-7

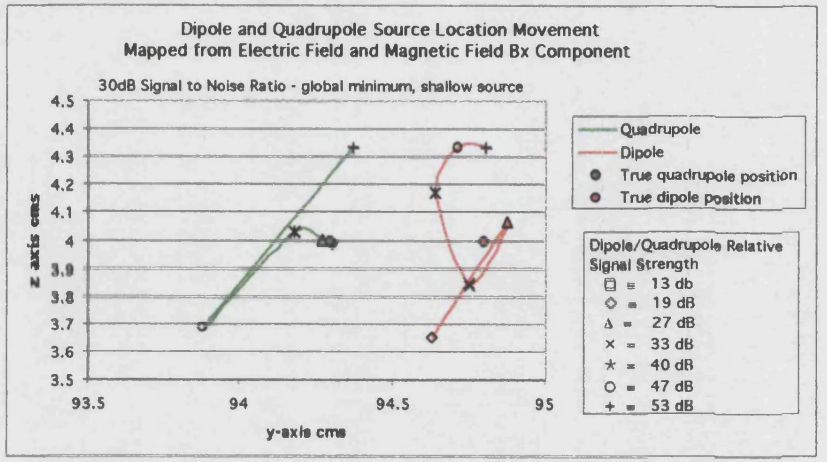


Fig 6c-8

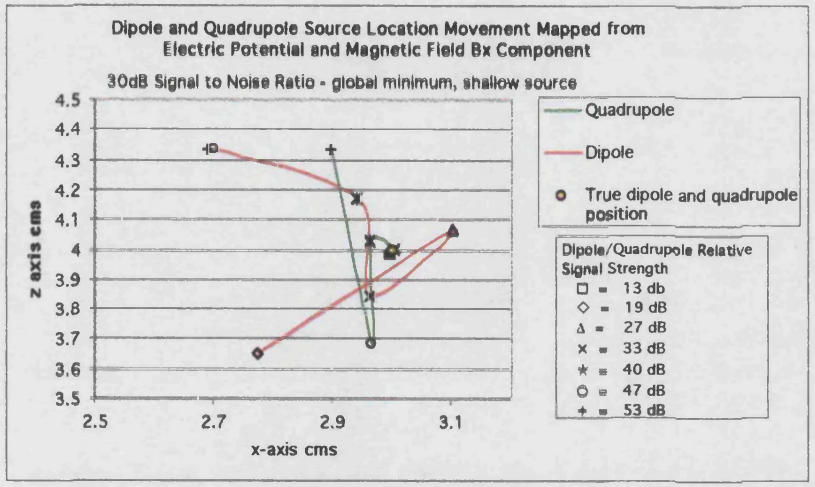


Fig 6c-9

Figs 6c-7 to 6c-9
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface electric potential and magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Mean location from 40 to 50 data sets.
Global minimum.

Electric Potential andMagnetic Field B_x Inverse

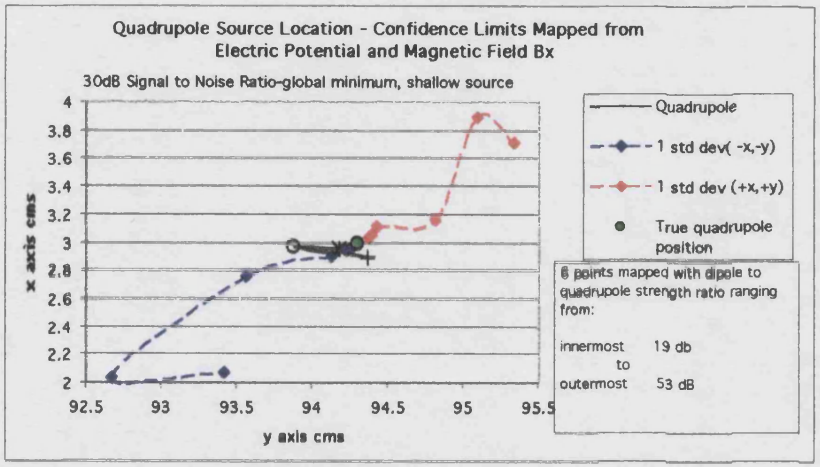


Fig 6c-10

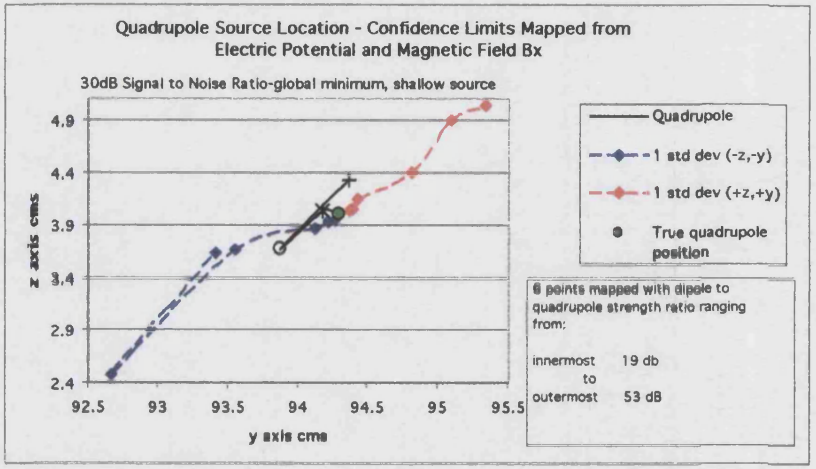


Fig 6c-11

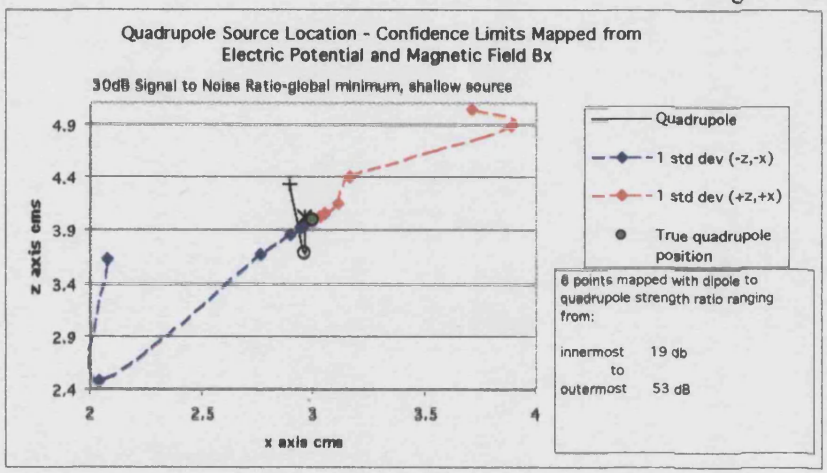


Fig 6c-12

Figs 6c-10 to 6c-12
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface electric potential and magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.
Global minimum.

Electric Potential andMagnetic Field B_x Inverse

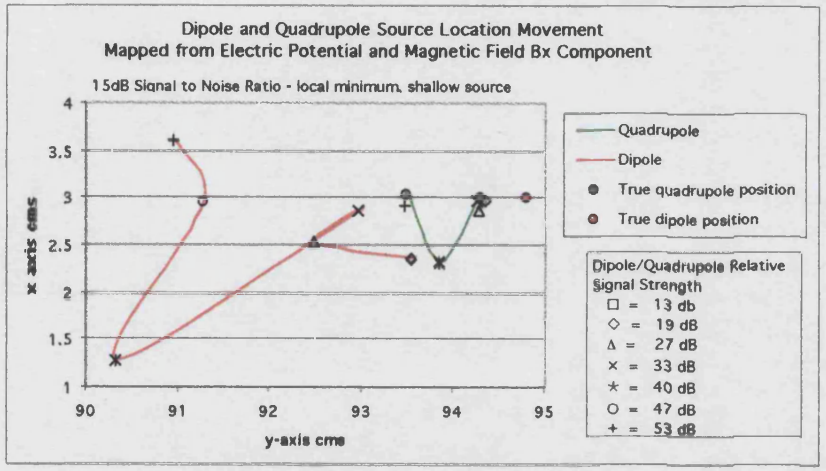


Fig 6c-13

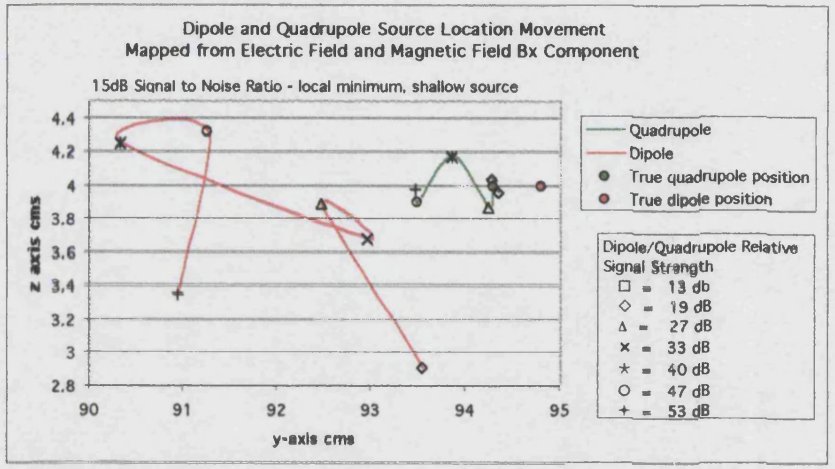


Fig 6c-14

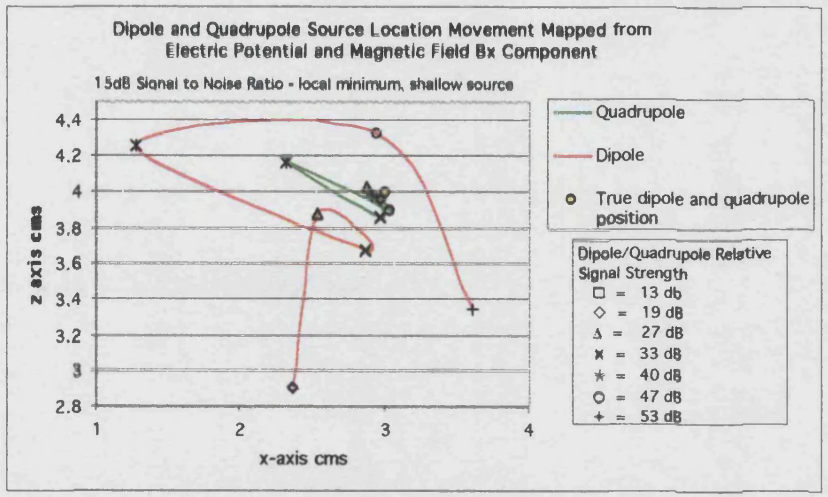


Fig 6c-15

Figs 6c-13 to 6c-15
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface electric potential and magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Mean location from 40 to 50 data sets.
Local minimum.

26/7/99

Electric Potential andMagnetic Field B_x Inverse

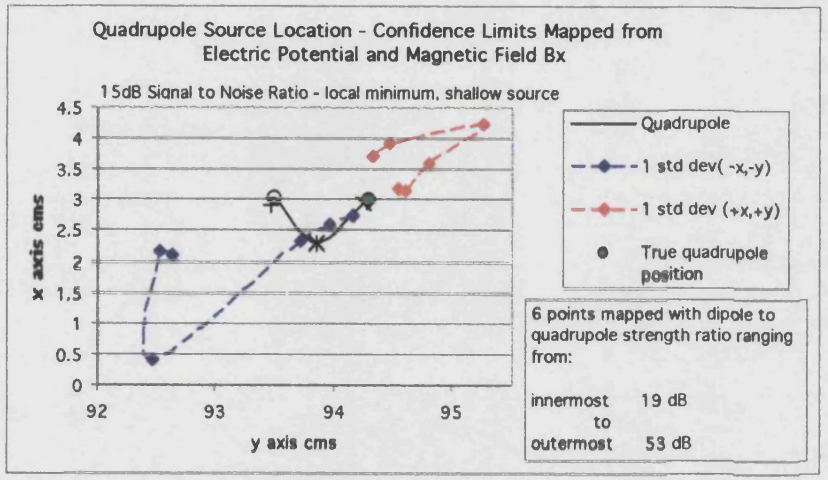


Fig 6c-16

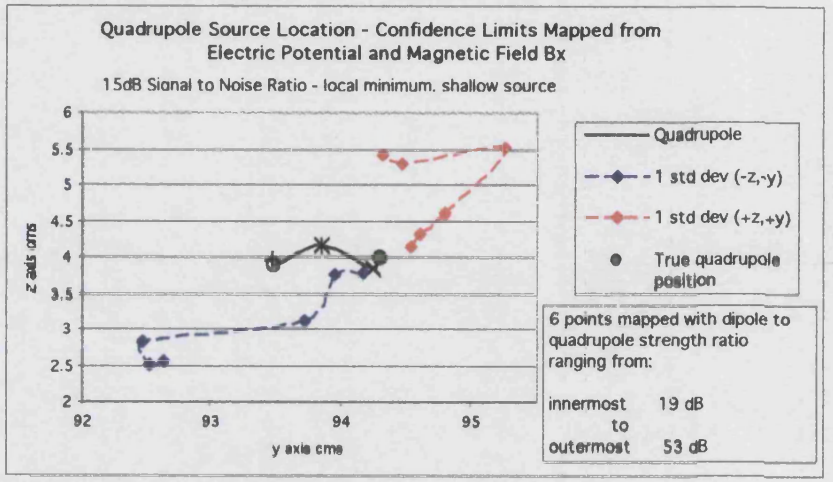


Fig 6c-17

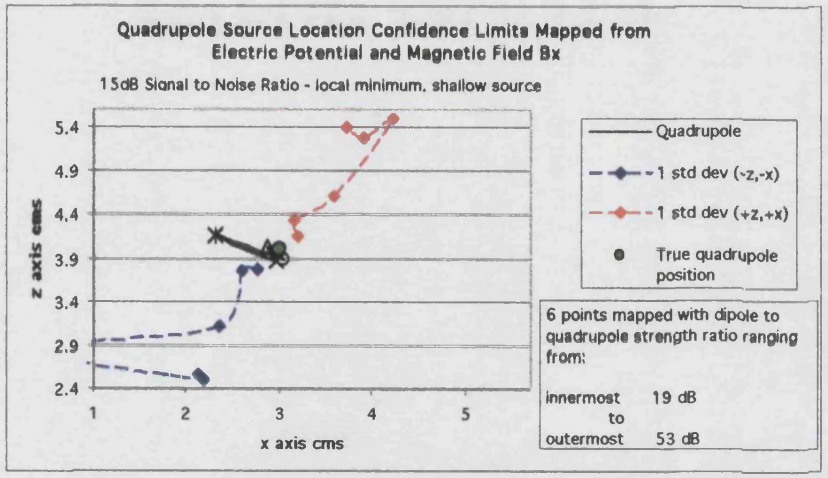


Fig 6c-18

Figs 6c-16 to 6c-18
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface electric potential and magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Standard deviation from 40 to 50 data sets.
Local minimum.

26/7/99

Electric Potential andMagnetic Field B_x Inverse

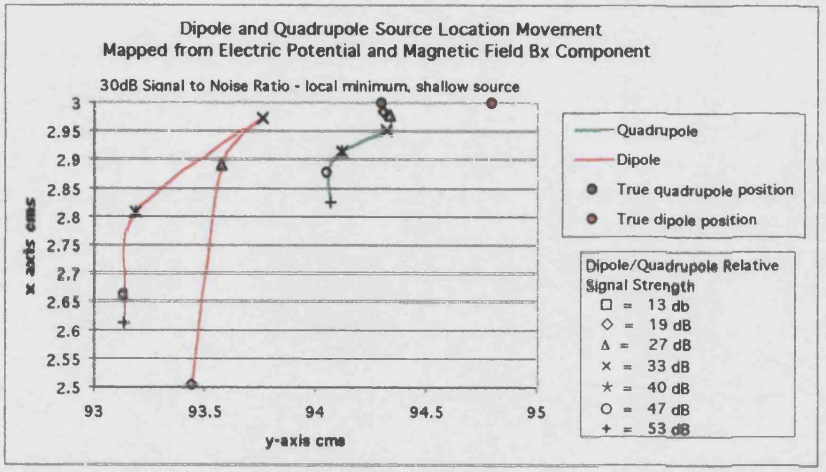


Fig 6c-19

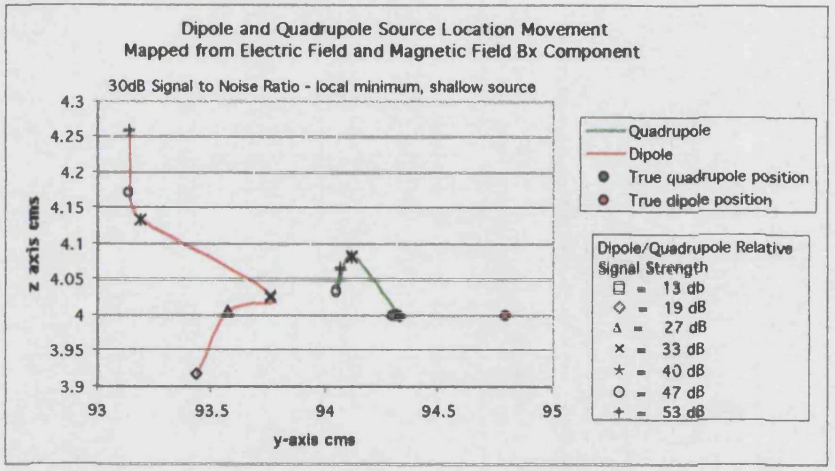


Fig 6c-20

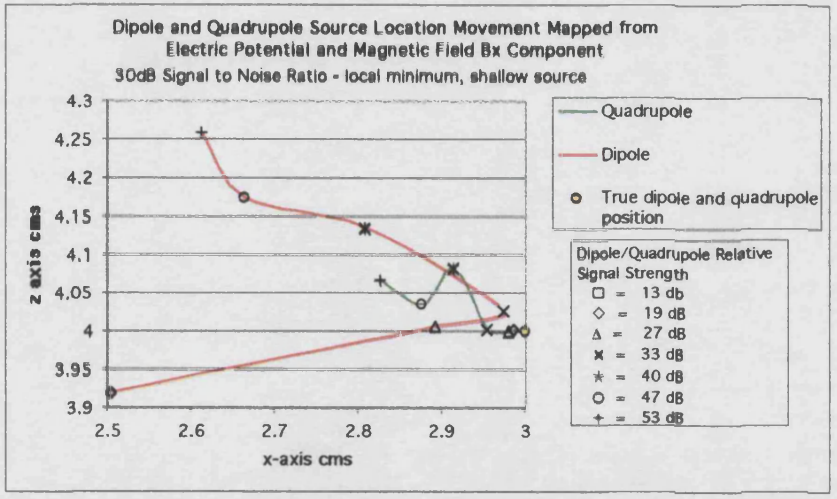


Fig 6c-21

Figs 6c-19 to 6c-21
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface electric potential and magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Mean location from 40 to 50 data sets.
Local minimum.

Electric Potential andMagnetic Field B_x Inverse

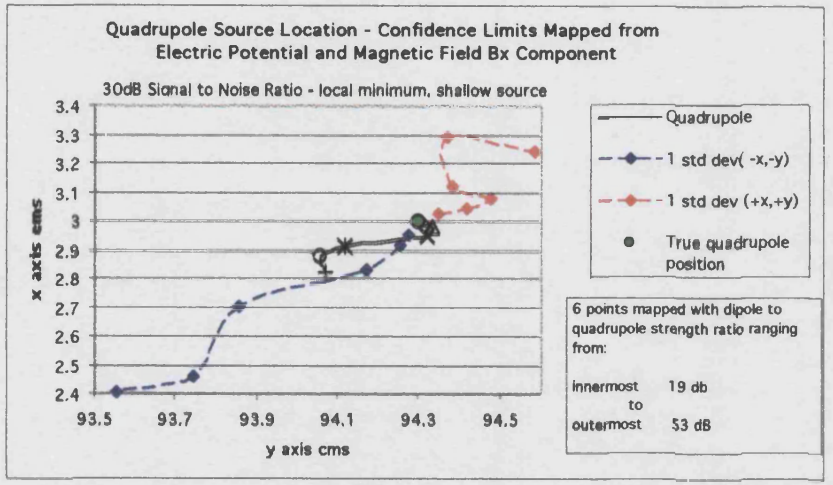


Fig 6c-22

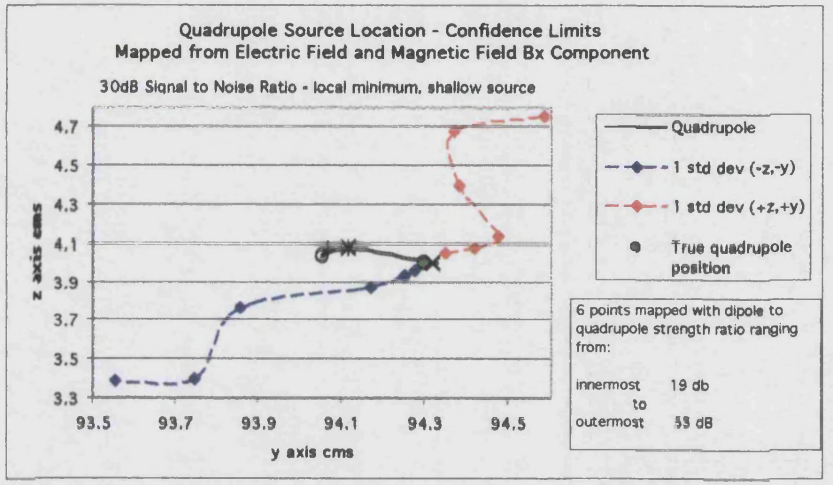


Fig 6c-23

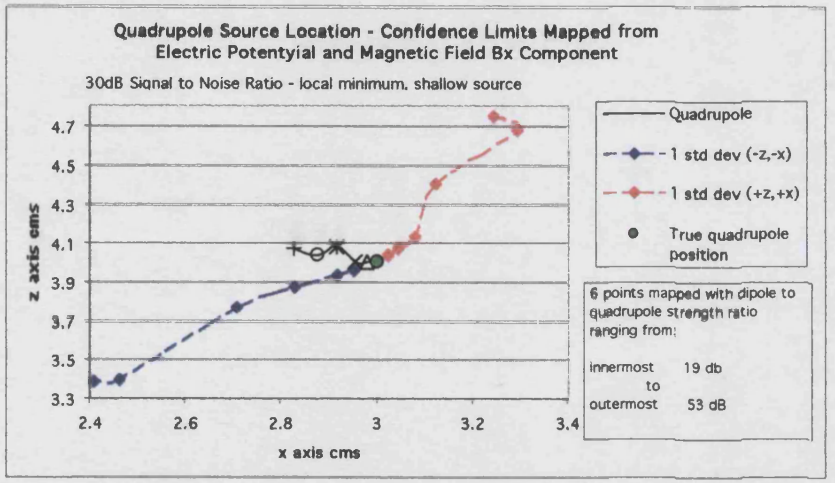


Fig 6c-24

Figs 6c-22 to 6c-24
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface electric potential and magnetic field B_x.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.
Local minimum.

26/7/99

Electric Potential and Magnetic Fields B_x and B_y Inverse Appendix 6d

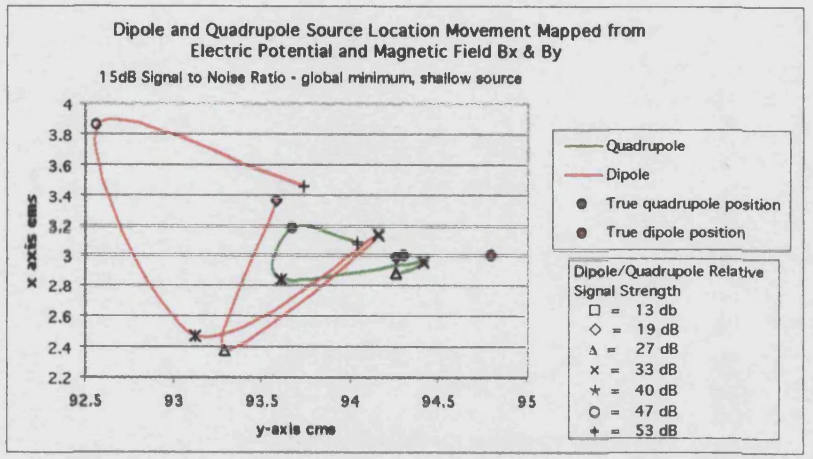


Fig 6d-1

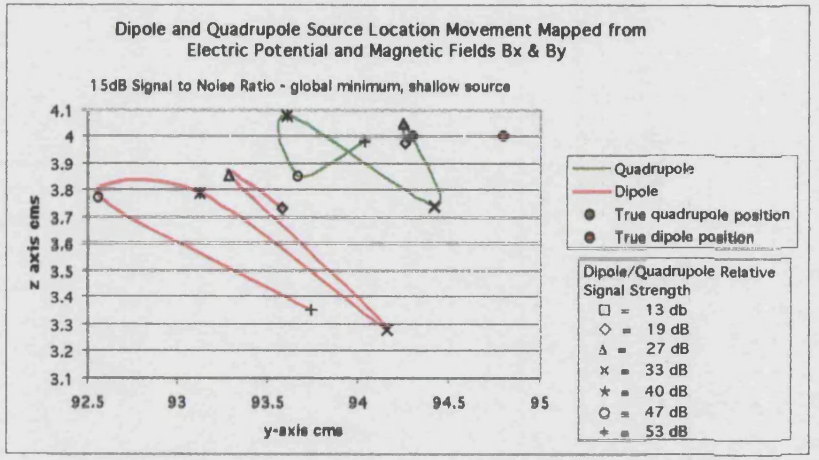


Fig 6d-2

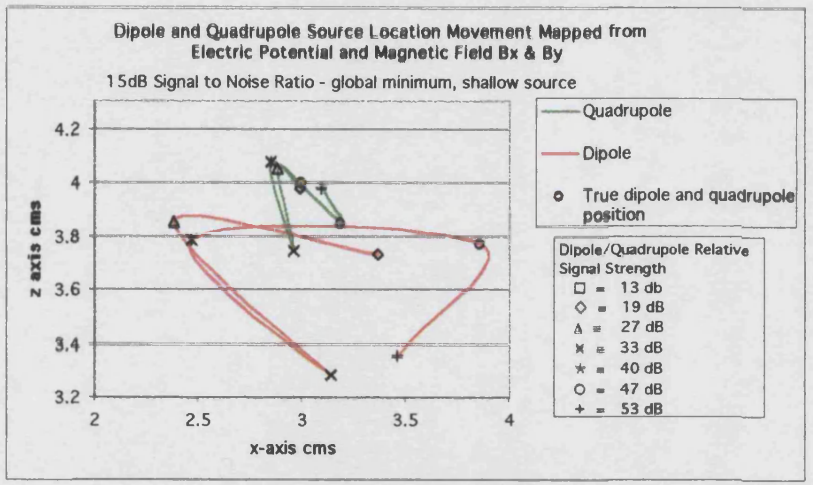


Fig 6d-3

Figs 6d-1 to 6d-3
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface electric potential and magnetic fields B_x and B_y.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Mean location from 40 to 50 data sets.
Global minimum.

26/7/99

Electric Potential and Magnetic Fields B_x and B_y Inverse

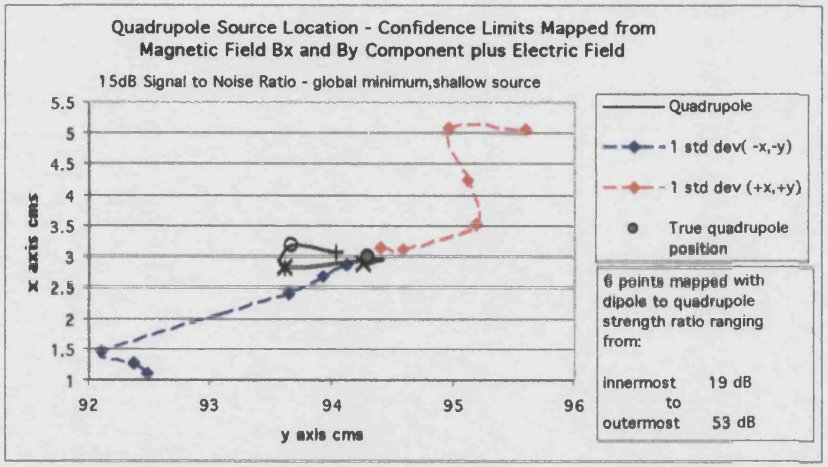


Fig 6d-4

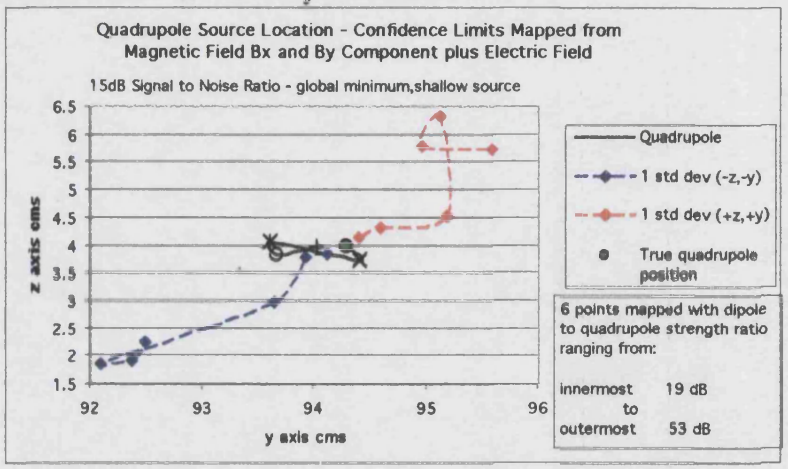


Fig 6d-5

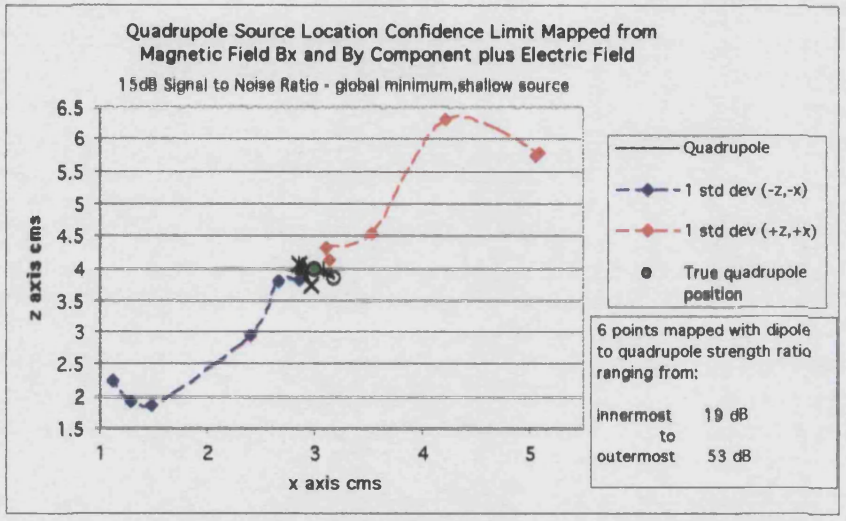


Fig 6d-6

Figs 6d-4 to 6d-6
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface electric potential and magnetic fields B_x and B_y.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Standard deviation from 40 to 50 data sets.
Global minimum

Electric Potential and Magnetic Fields B_x and B_y Inverse

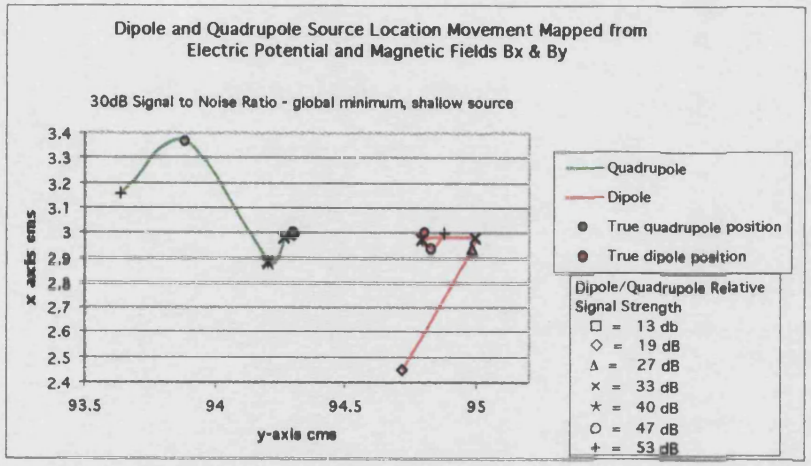


Fig 6d-7

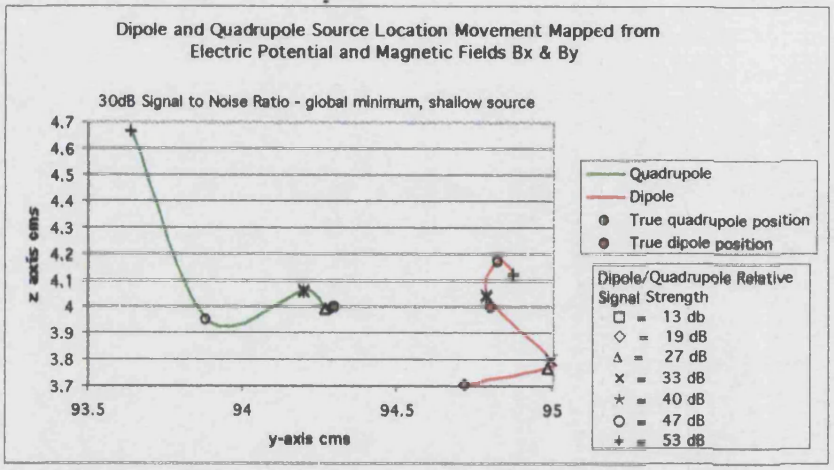


Fig 6d-8

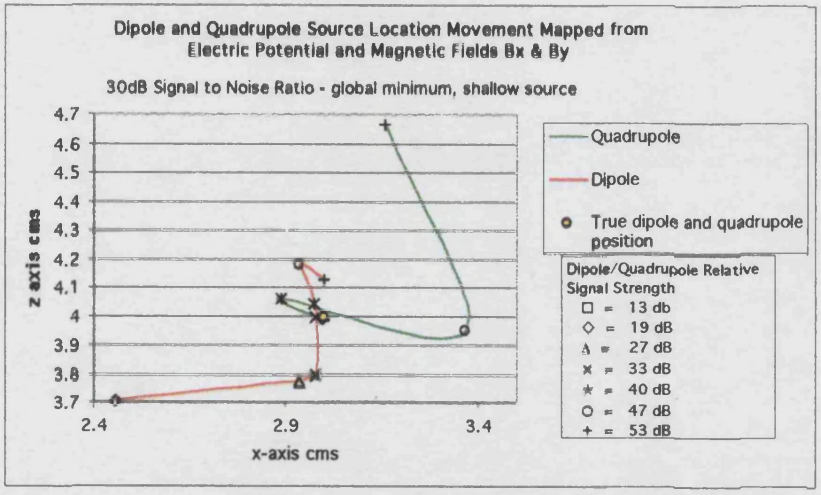


Fig 6d-9

Figs 6d-7 to 6d-9
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface electric potential and magnetic fields B_x and B_y.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Mean location from 40 to 50 data sets.
Global minimum.

Electric Potential and Magnetic Fields B_x and B_y Inverse

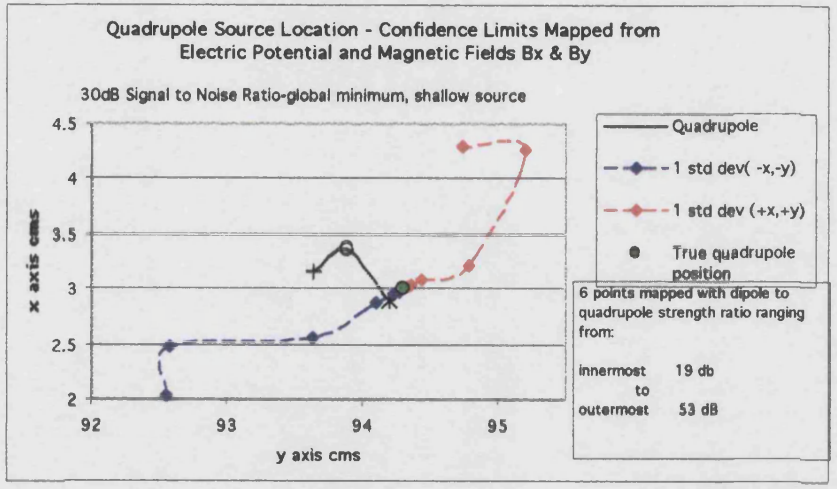


Fig 6d-10

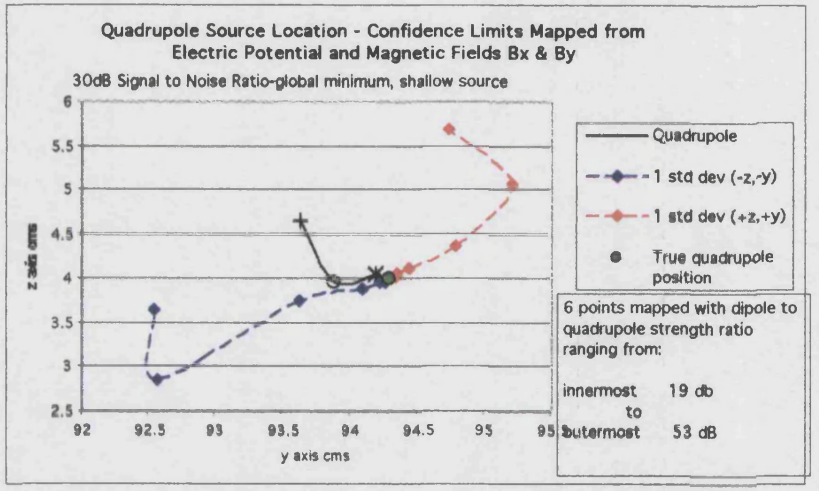


Fig 6d-11

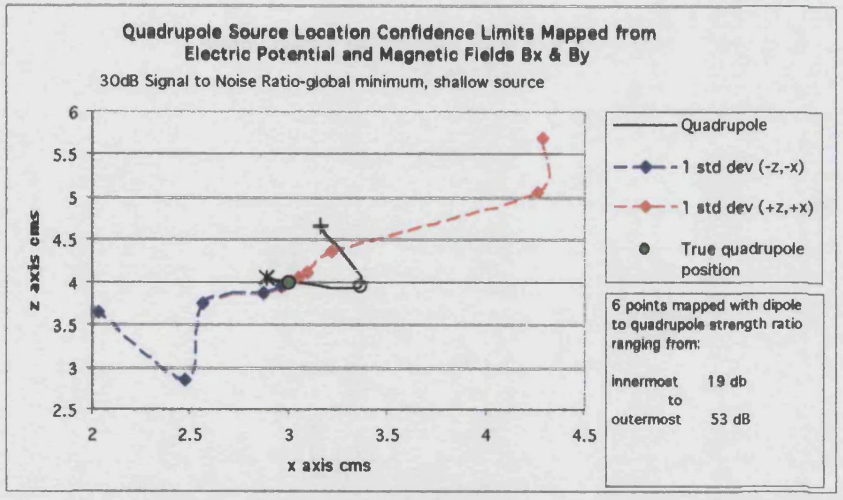


Fig 6d-12

Figs 6d-10 to 6d-12
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface electric potential and magnetic fields B_x and B_y.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.
Global minimum

Magnetic Field B_y Inverse

Appendix 6e

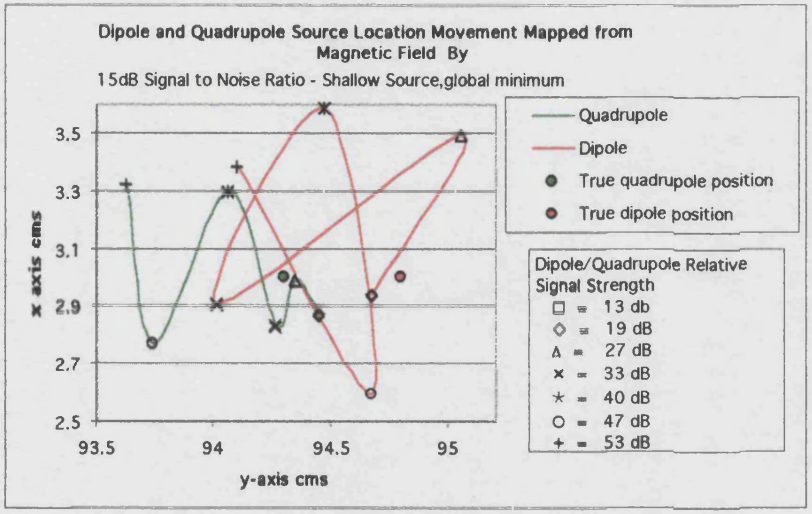


Fig 6e-1

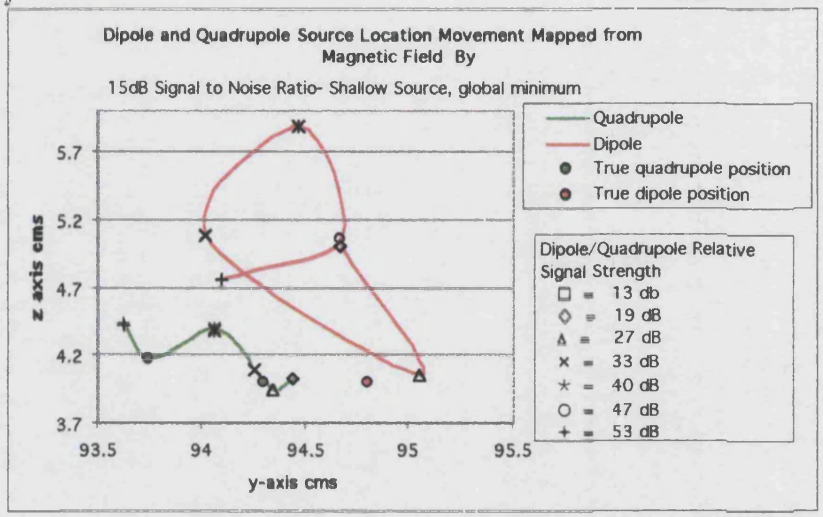


Fig 6e-2

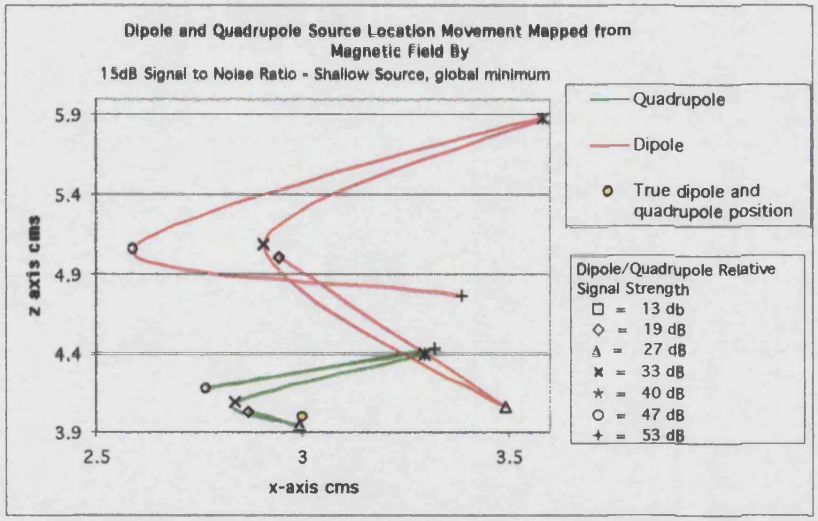


Fig 6e-3

Figs 6e-1 to 6e-3
x vs y, z vs y and z vs x plots
Inverse location of a dipole and a quadrupole point current source using the surface magnetic field B_y.
Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Mean location from 40 to 50 data sets.
Global minimum.

Magnetic Field B_y Inverse

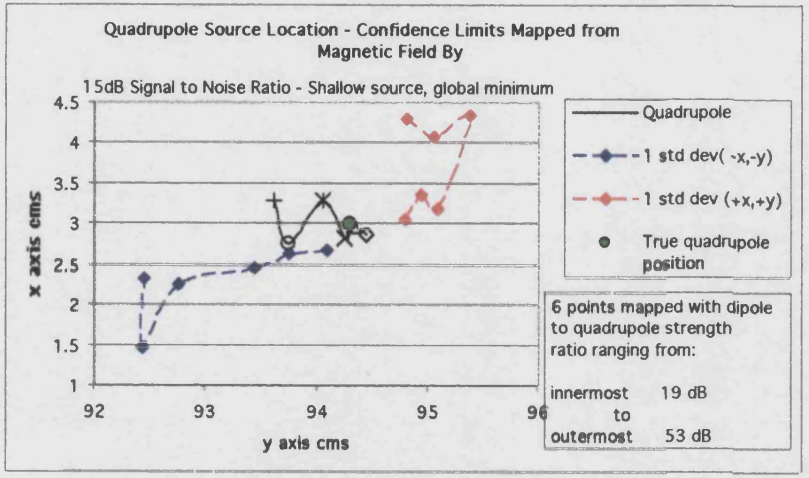


Fig 6e-4

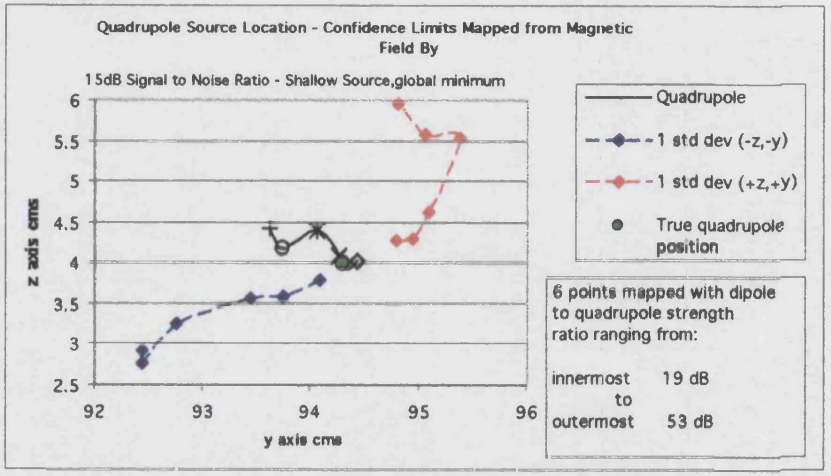


Fig 6e-5

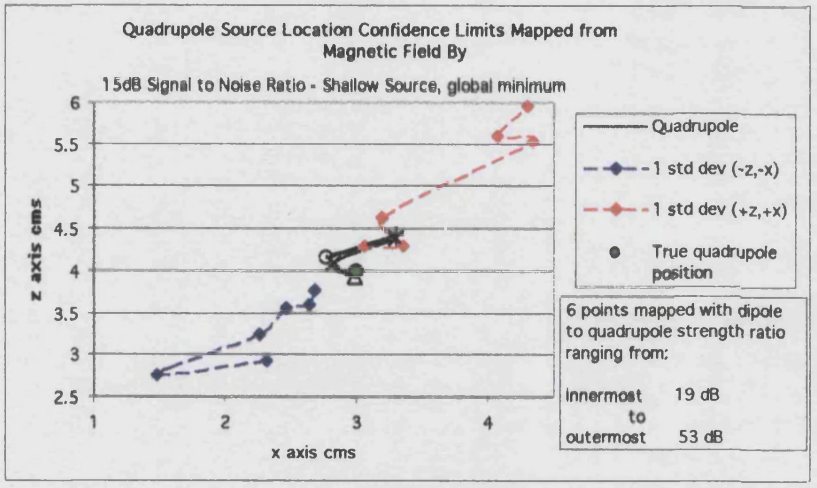


Fig 6e-6

Figs 6e-4 to 6e-6
x vs y, z vs y and z vs x plots
One standard deviation from the above inverse location of the quadrupole point current source using the surface magnetic field B_y. Dipole at (3,94.8,4) and quadrupole at (3,94.3,4)
Noise at 15 dB SNR
Standard deviation from 40 to 50 data sets.
Global minimum.

Confidence Limits Comparison of the Unified Models – 15dB Case

Appendix 6f

x-y comparison

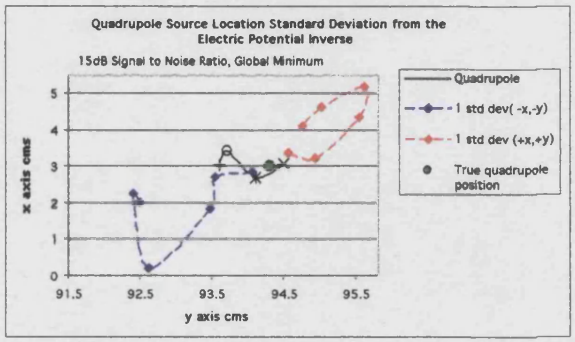


Fig 6f-1

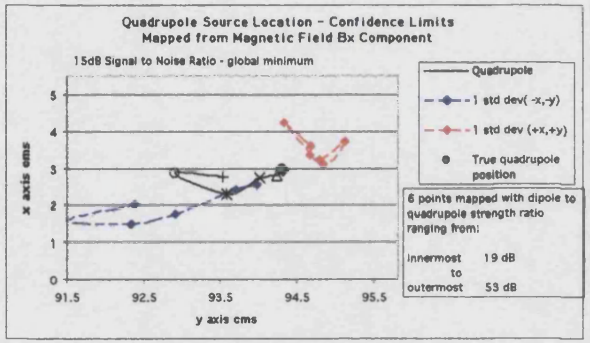


Fig 6f-2

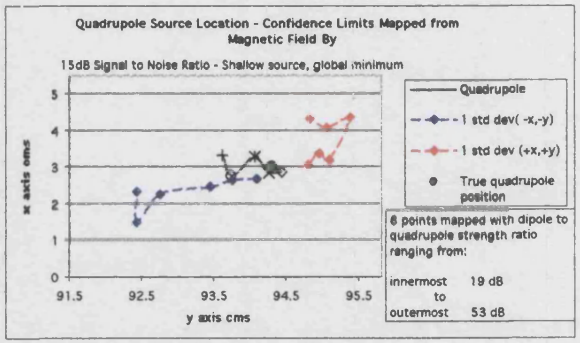


Fig 6f-3

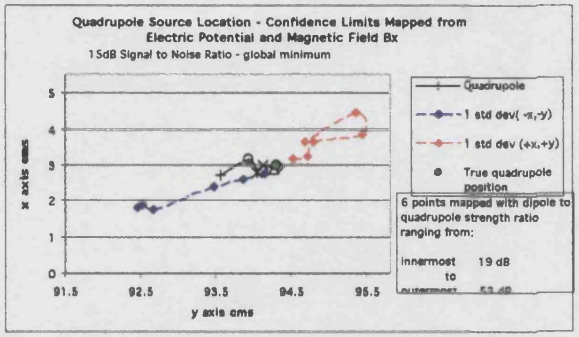


Fig 6f-4

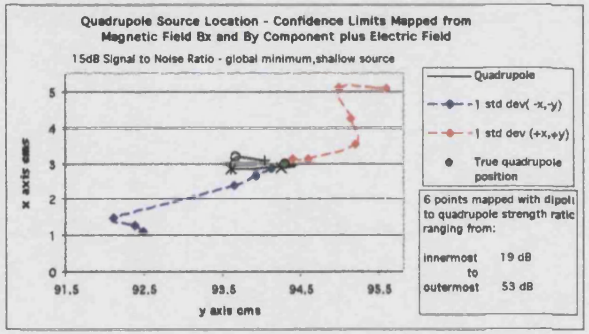


Fig 6f-6

z-y comparison

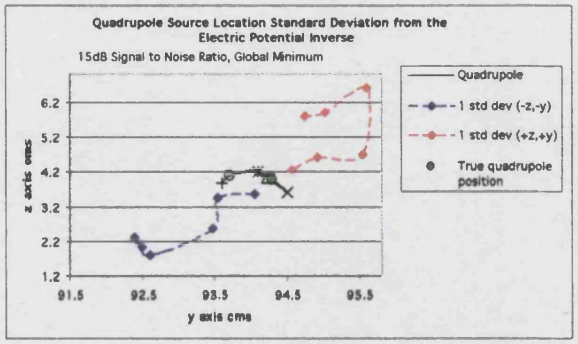


Fig 6f-6

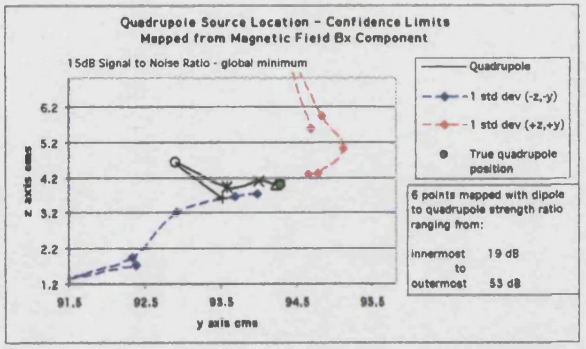


Fig 6f-7

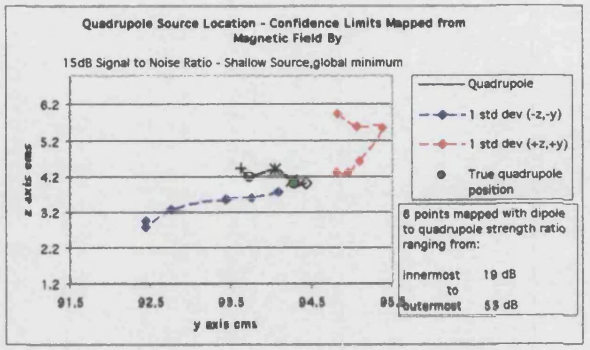


Fig 6f-8

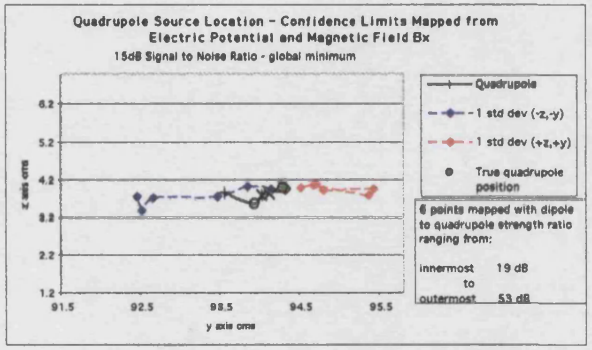


Fig 6f-9

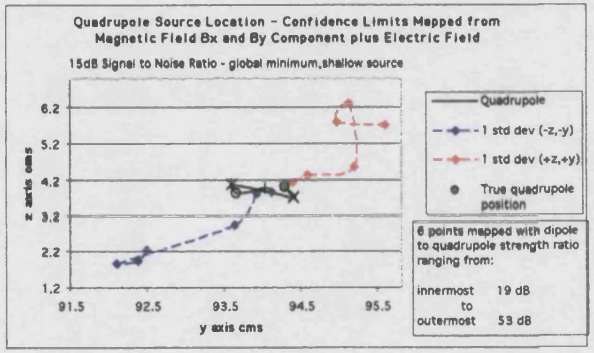


Fig 6f-10

A6f-2

Confidence Limits Comparison of the Unified Models – 30dB Case

Appendix 6g

x-y comparison

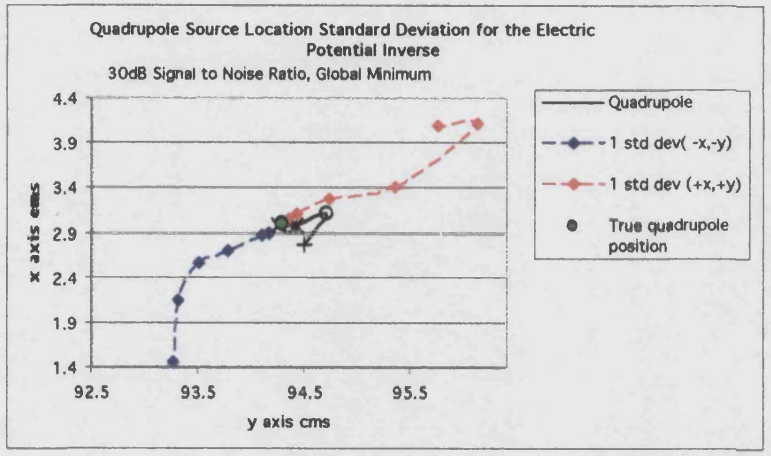


Fig 6g-1

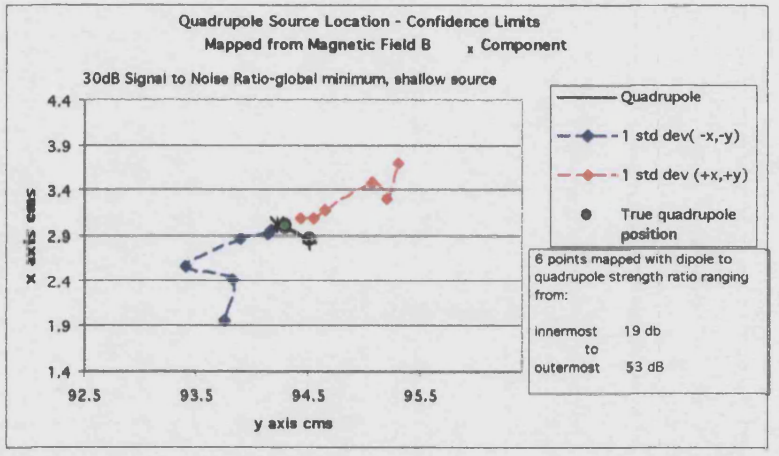


Fig 6g-2

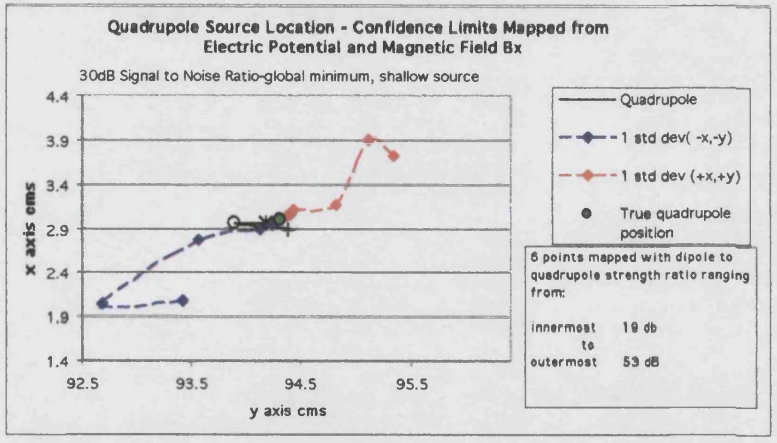


Fig 6g-3

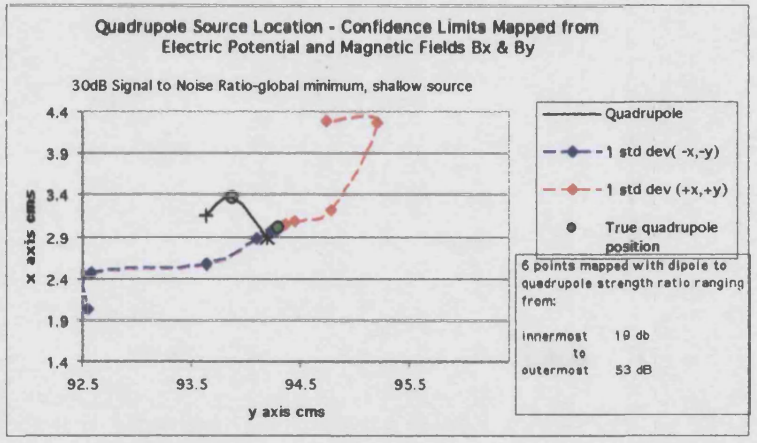


Fig 6g-4

z-y comparison

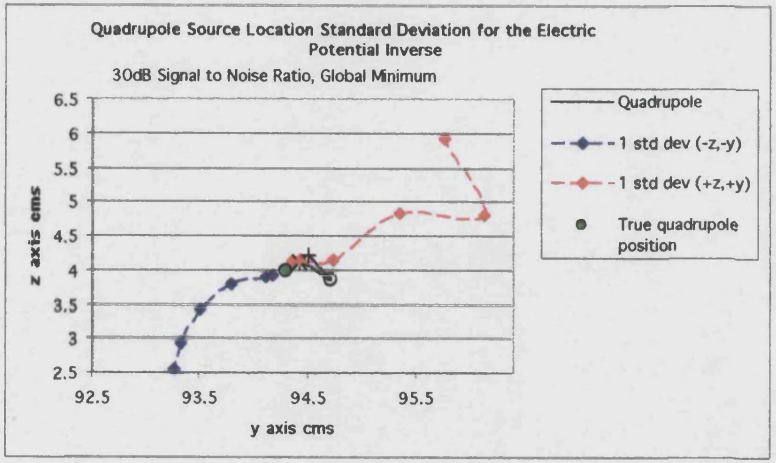


Fig 6g-5

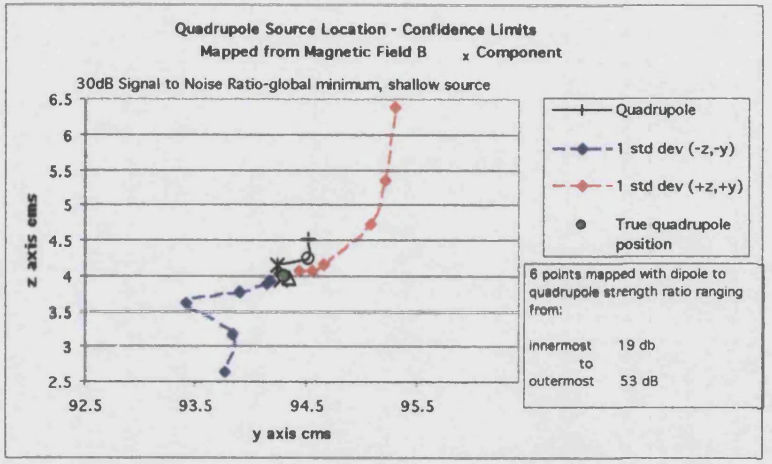


Fig 6g-6

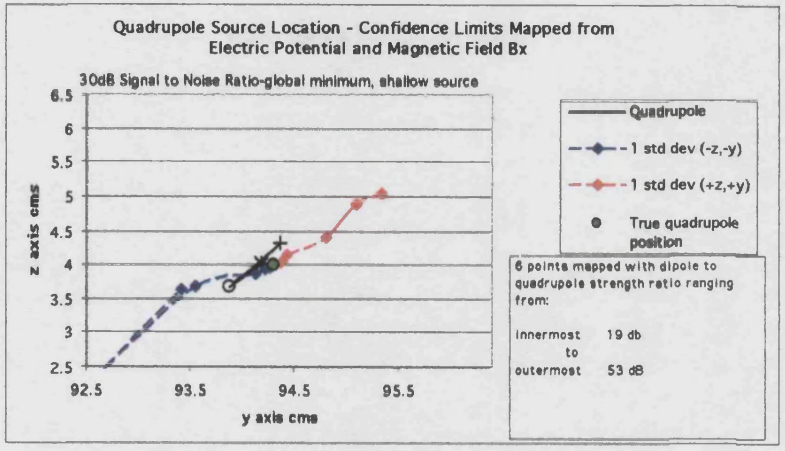


Fig 6g-7

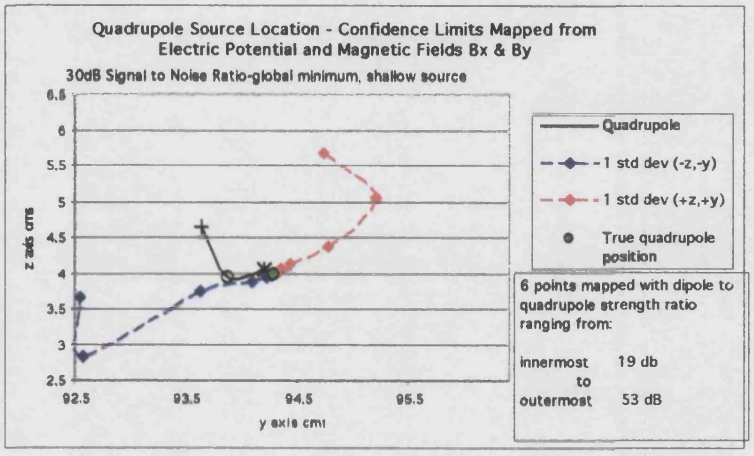


Fig 6g-8

Appendix 7. Sensitivity to Source Depth

Sensitivity of the Electric Potential Inverse to Source Depth

Appendix7a

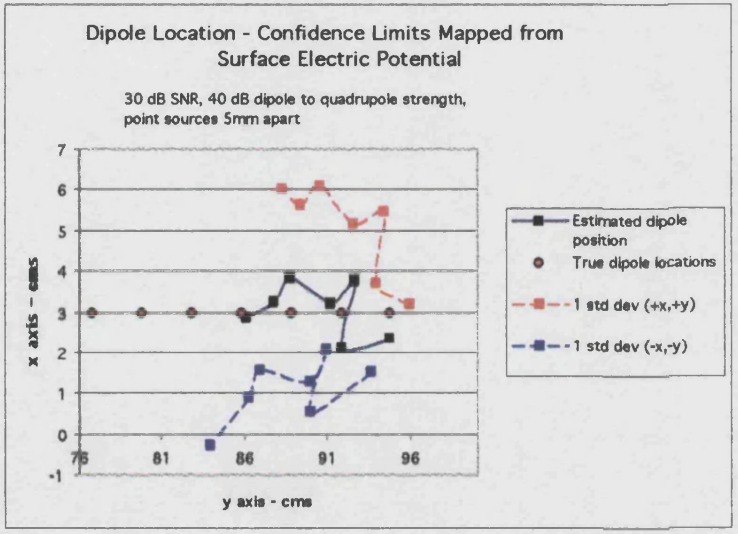


Fig 7a-1

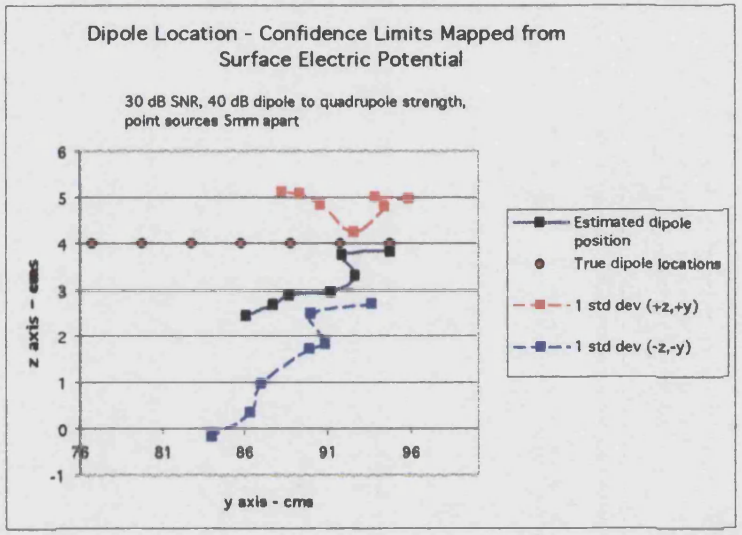


Fig 7a-2

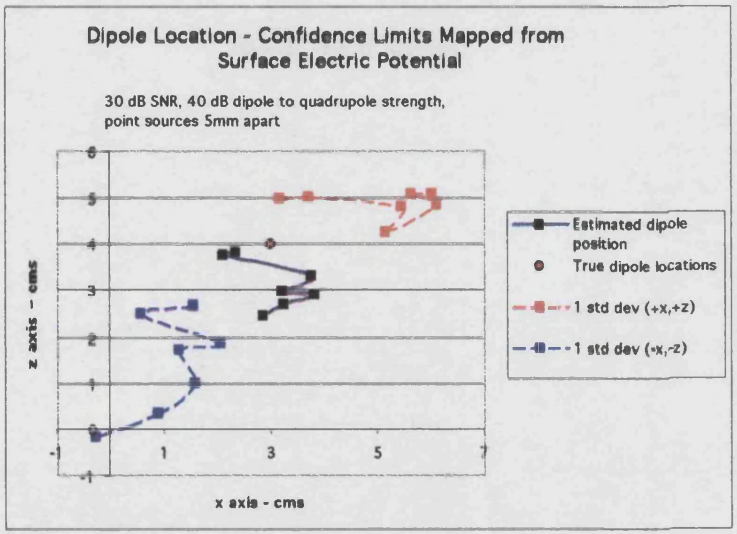


Fig 7a-3

Figs 7a-1 to 7a-3
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the dipole point current source using the surface electric potential.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.

Sensitivity of the Electric Potential Inverse to Source Depth

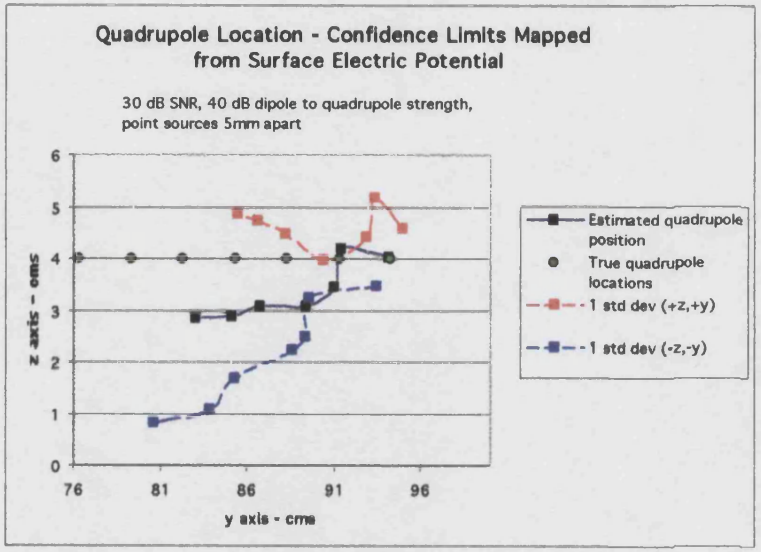


Fig 7a-4

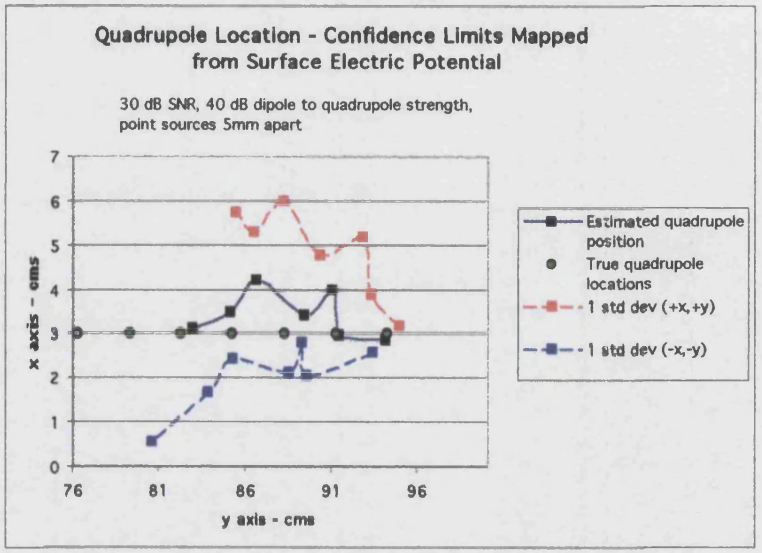


Fig 7a-5

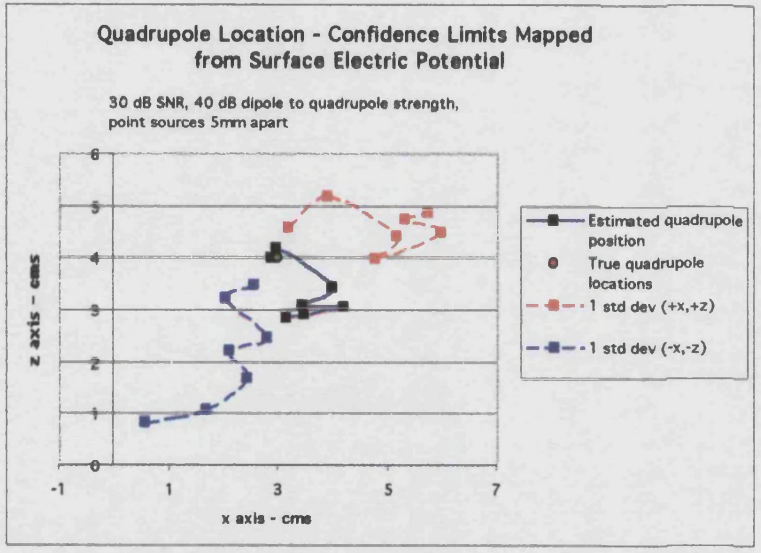


Fig 7a-6

Figs 7a-4 to 7a-6
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the quadrupole point current source using the surface electric potential.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.

Sensitivity of the Magnetic Field B_x and B_y Inverse to Source Depth

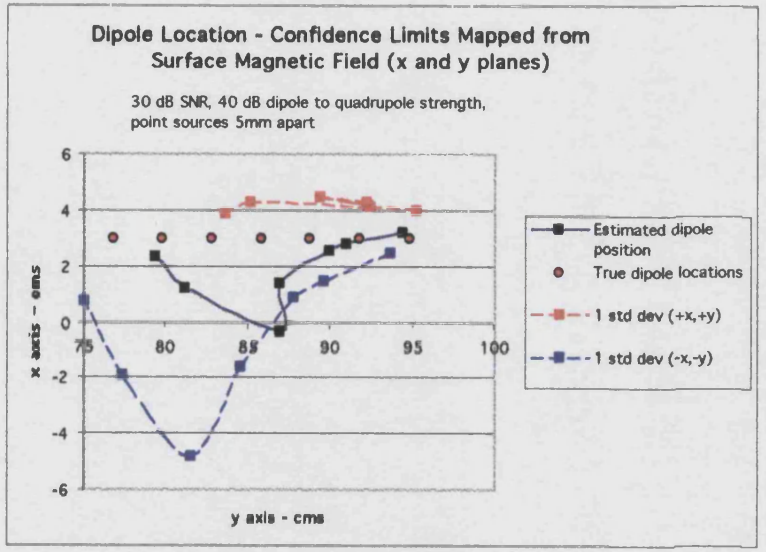


Fig 7a-7

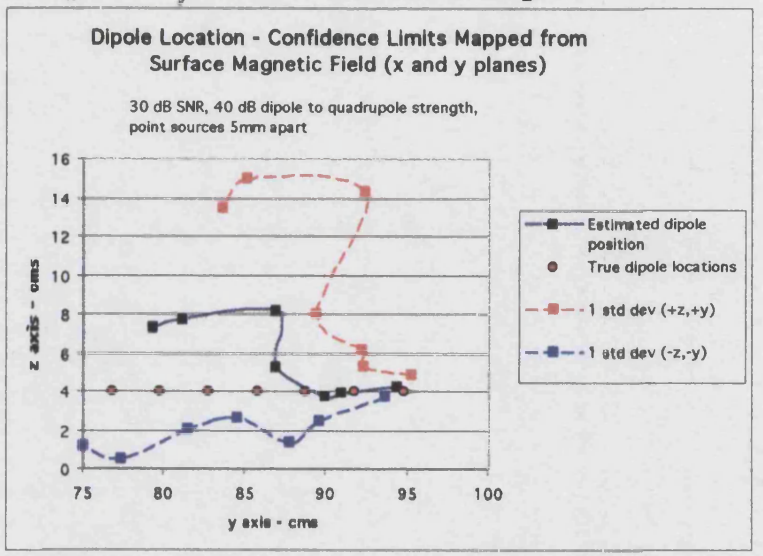


Fig 7a-8

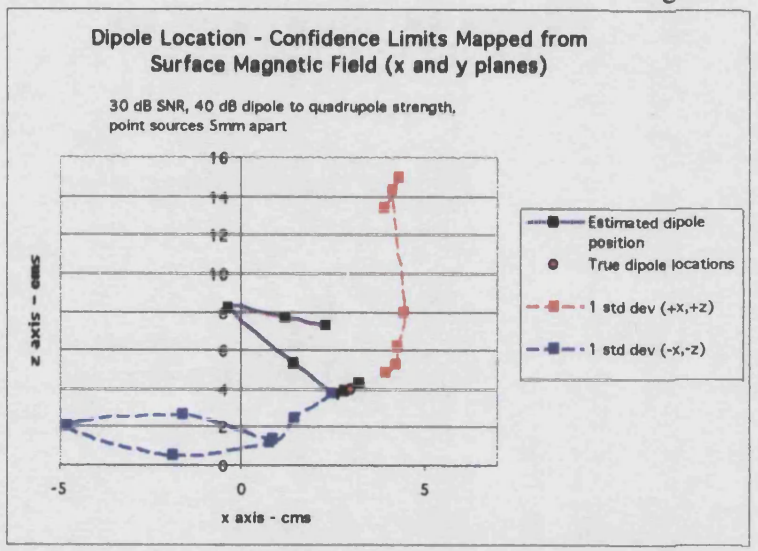


Fig 7a-9

Figs 7a-7 to 7a-9
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the dipole point current source using the surface magnetic fields Bx and By.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.

Sensitivity of the Magnetic Field B_x and B_y Inverse to Source Depth

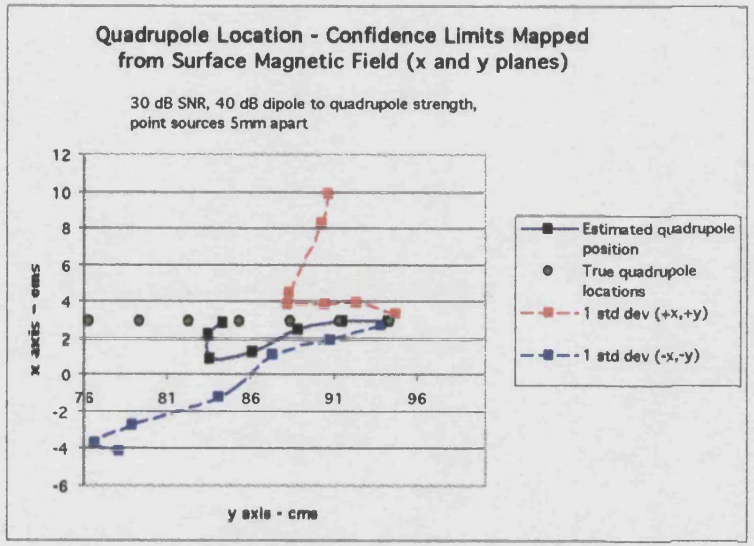


Fig 7a-10

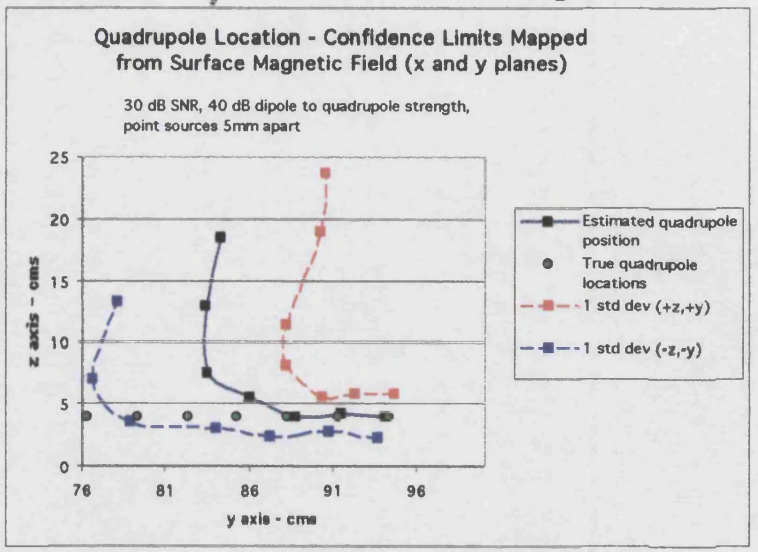


Fig 7a-11

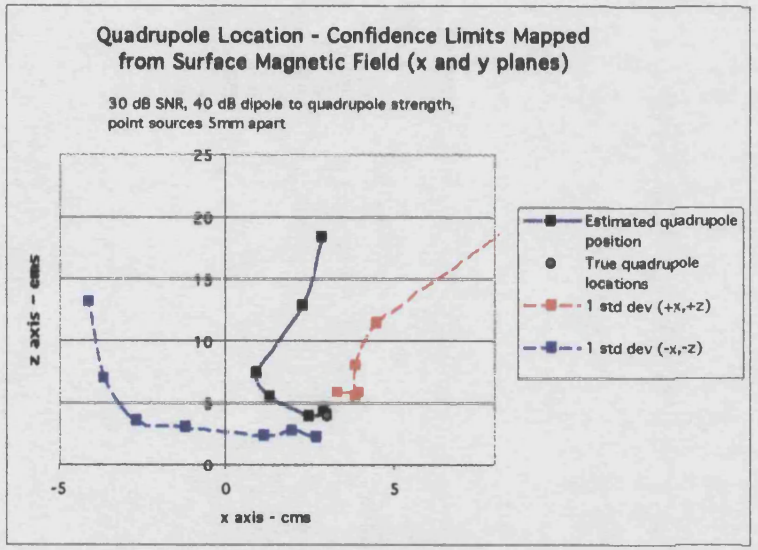


Fig 7a-12

Figs 7a-10 to 7a-12
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the quadrupole point current source using the surface magnetic fields Bx and By.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.

Sensitivity of the Magnetic Field B_x and B_z Inverse to Source Depth

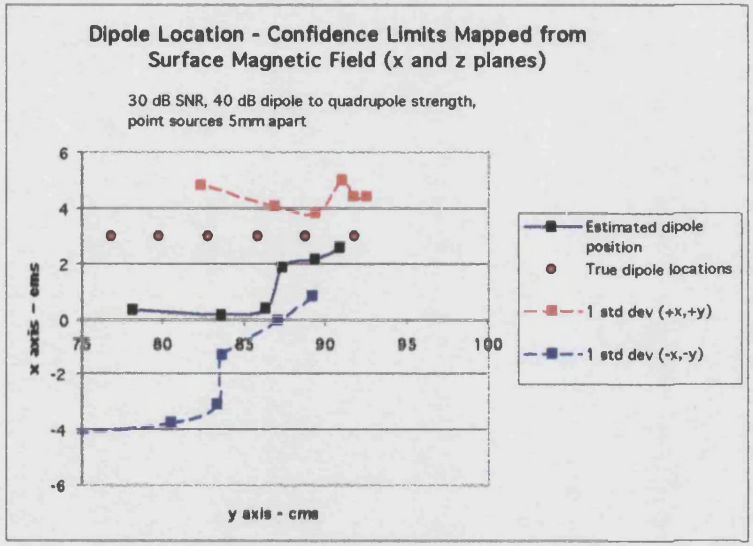


Fig 7a-13

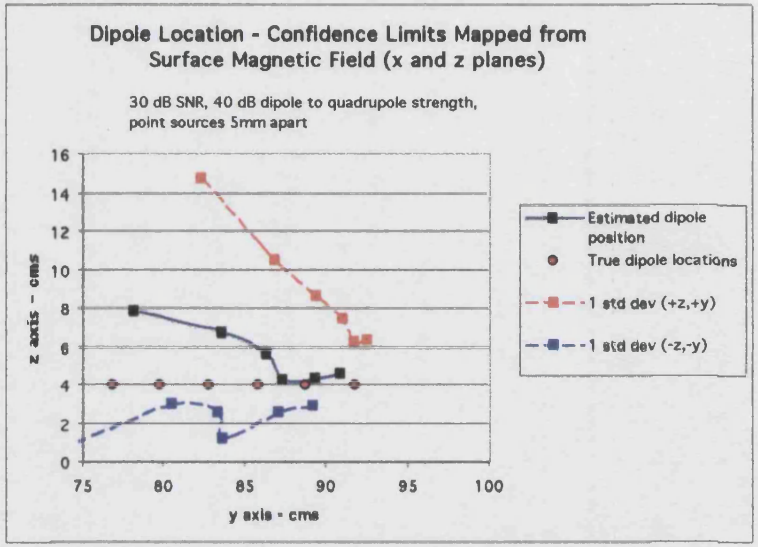


Fig 7a-14

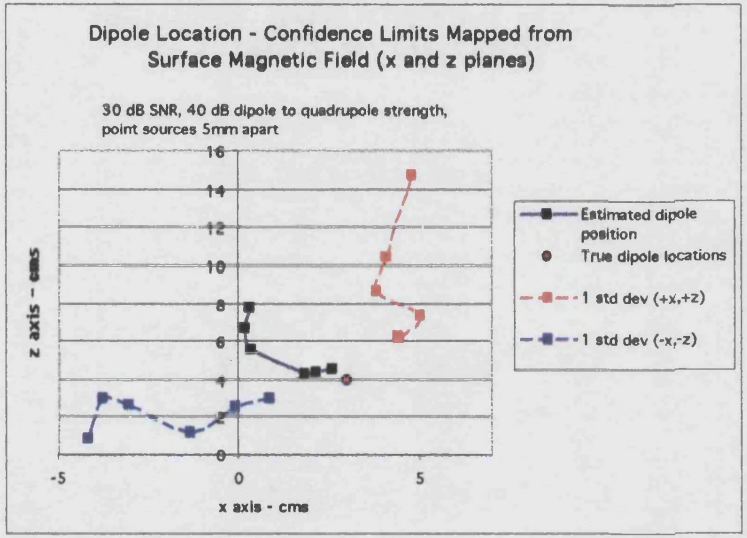


Fig 7a-15

Figs 7a-13 to 7a-15
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the dipole point current source using the surface magnetic fields Bx and Bz.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.

Sensitivity of the Magnetic Field B_x and B_z Inverse to Source Depth

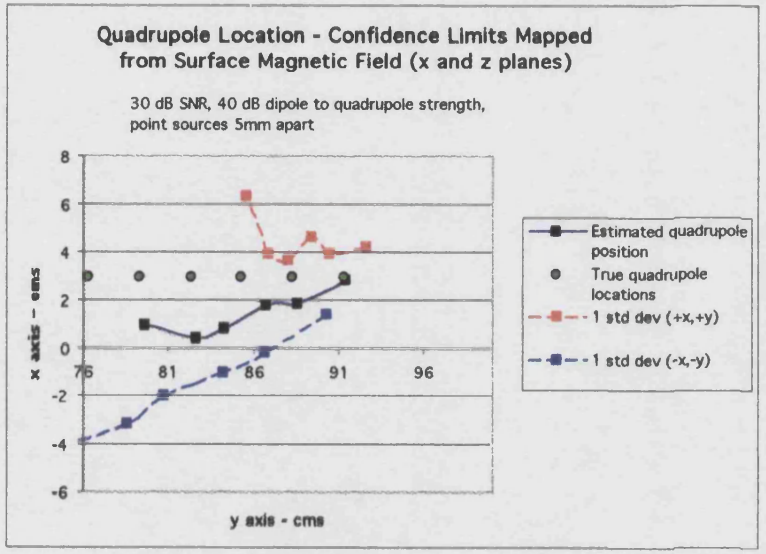


Fig 7a-16

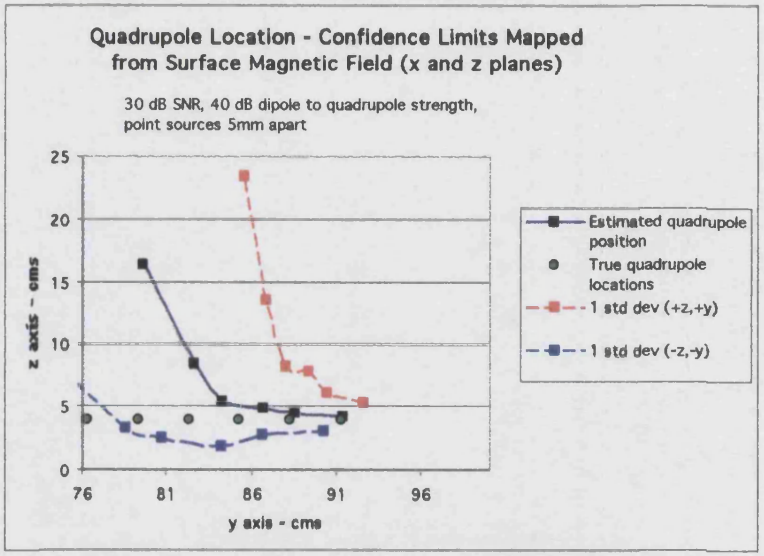


Fig 7a-17

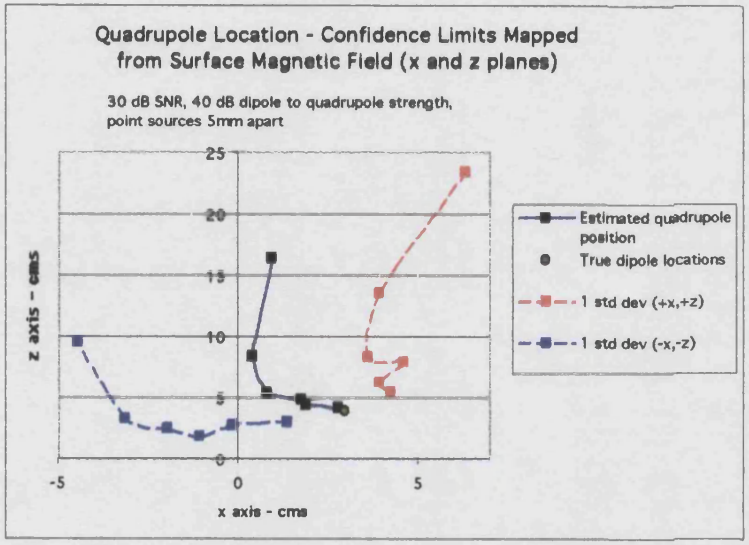


Fig 7a-18

Figs 7a-16 to 7a-18
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the quadrupole point current source using the surface magnetic fields Bx and Bz.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets

Sensitivity of the Electric Potential and Magnetic Fields B_x and B_y Inverse to Source Depth

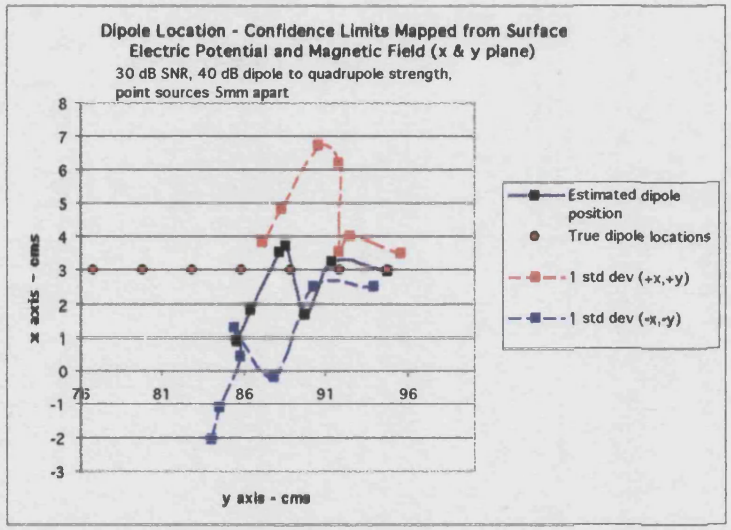


Fig 7a-19

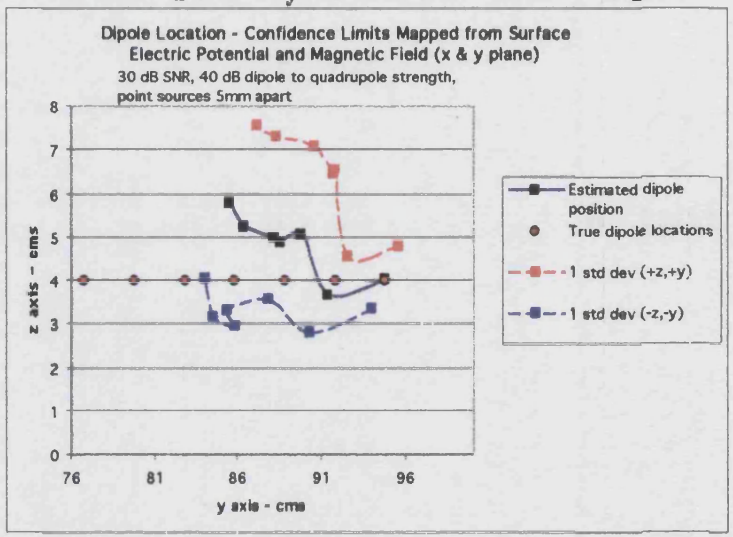


Fig 7a-20

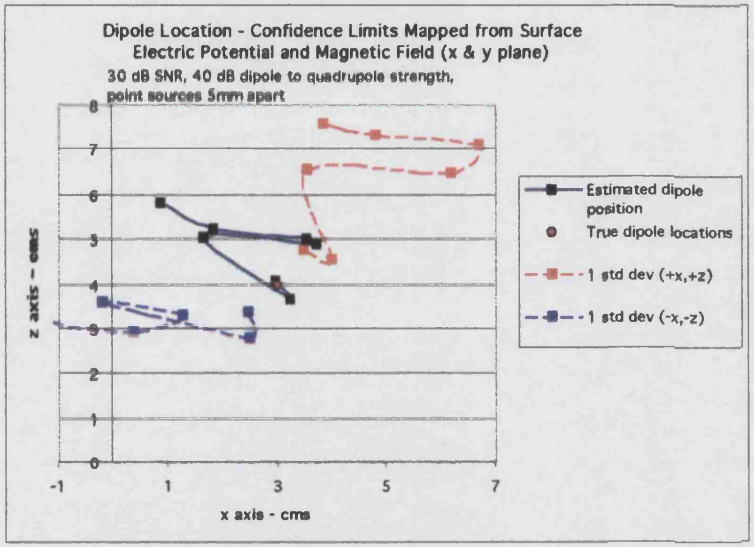


Fig 7a-21

Figs 7a-19 to 7a-21
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the dipole point current source using the surface electric potential and magnetic fields Bx znd By.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.

Sensitivity of the Electric Potential and Magnetic Fields B_x and B_y Inverse to Source Depth

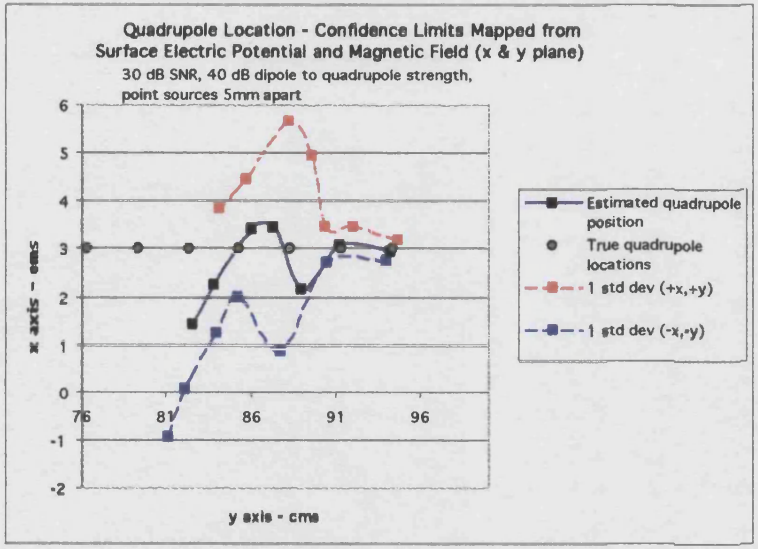


Fig 7a-22

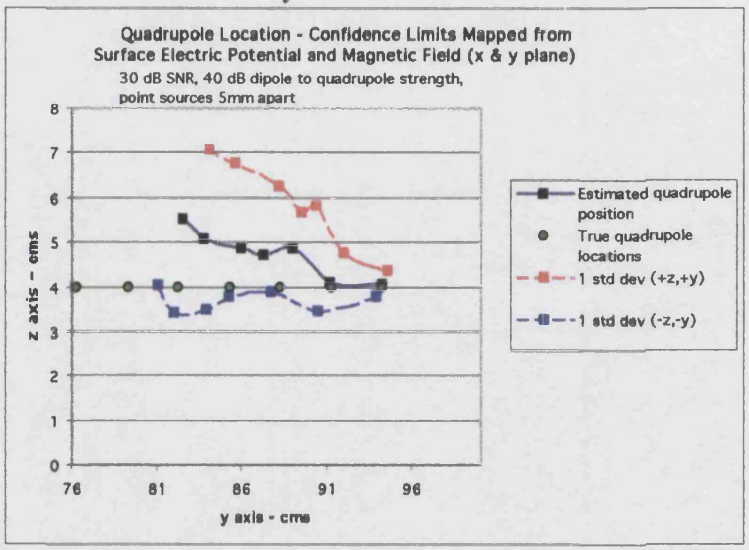


Fig 7a-23

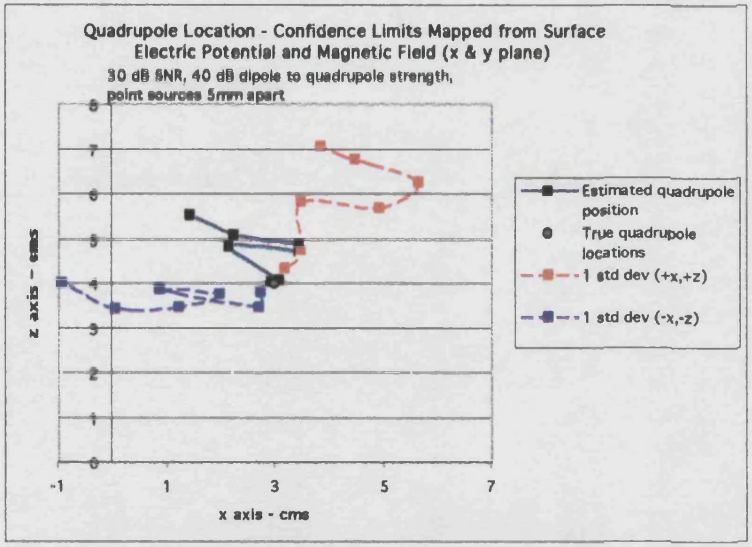


Fig 7a-24

Figs 7a-22 to 7a-24
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the quadrupole point current source using the surface electric potential and magnetic fields Bx znd By.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.

Sensitivity of the Magnetic Field B_y Inverse to Source Depth – Rank=10

Appendix7b

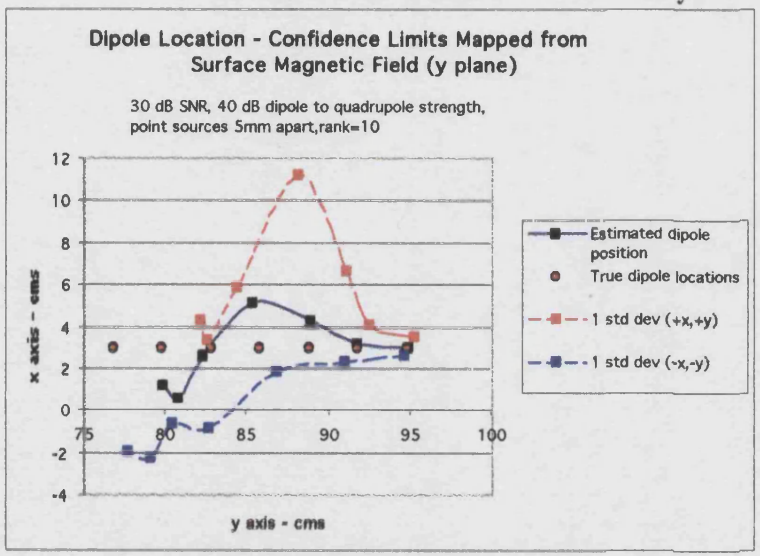


Fig 7b-1

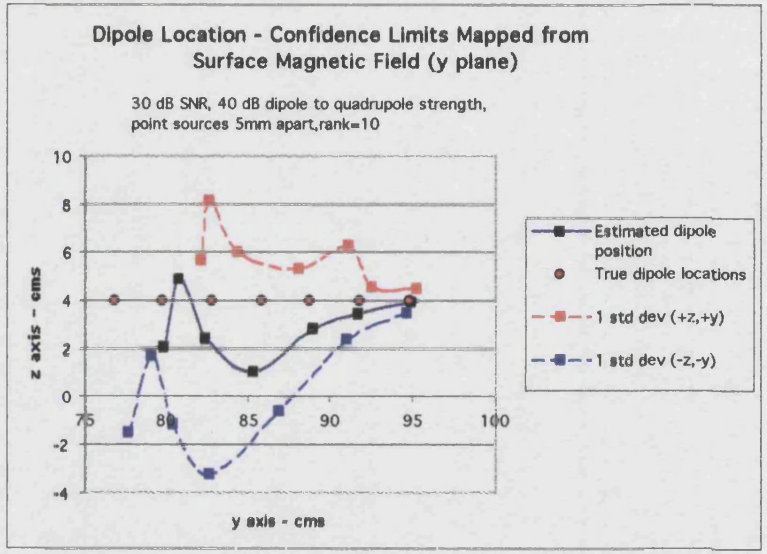


Fig 7b-2

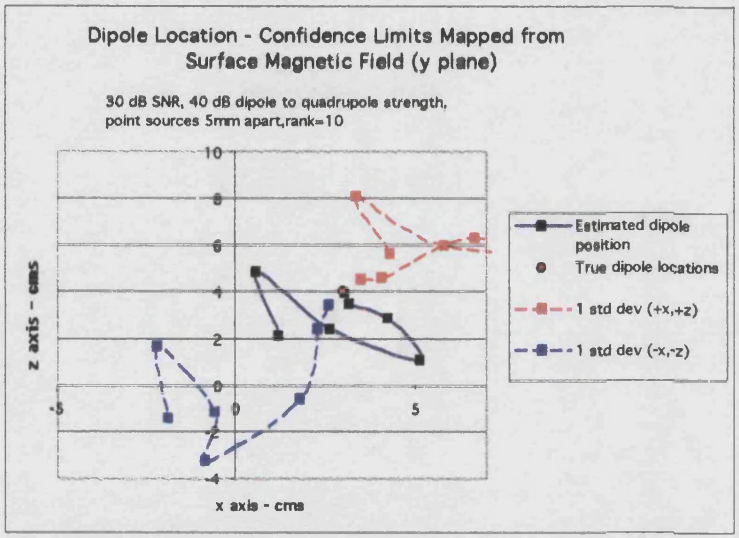


Fig 7b-3

Figs 7b-1 to 7b-3
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the dipole point current source using the surface magnetic field By. Rank deficient case rank=10.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.

Sensitivity of the Magnetic Field B_y Inverse to Source Depth – Rank=10

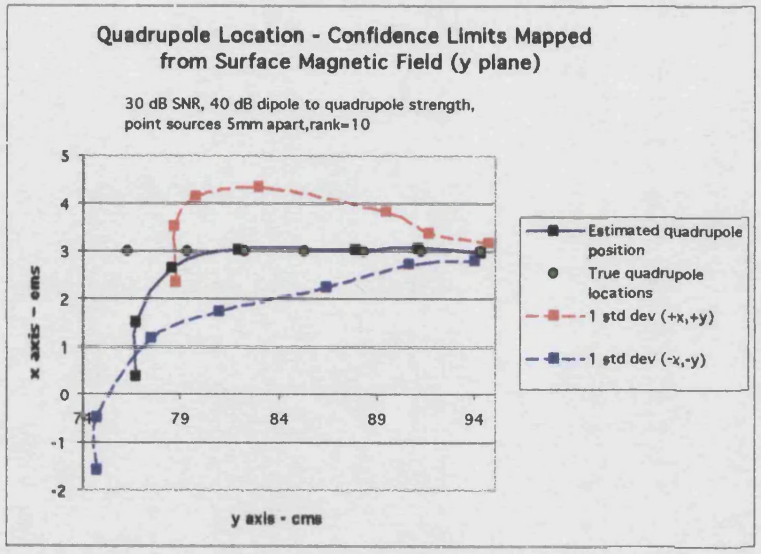


Fig 7b-4

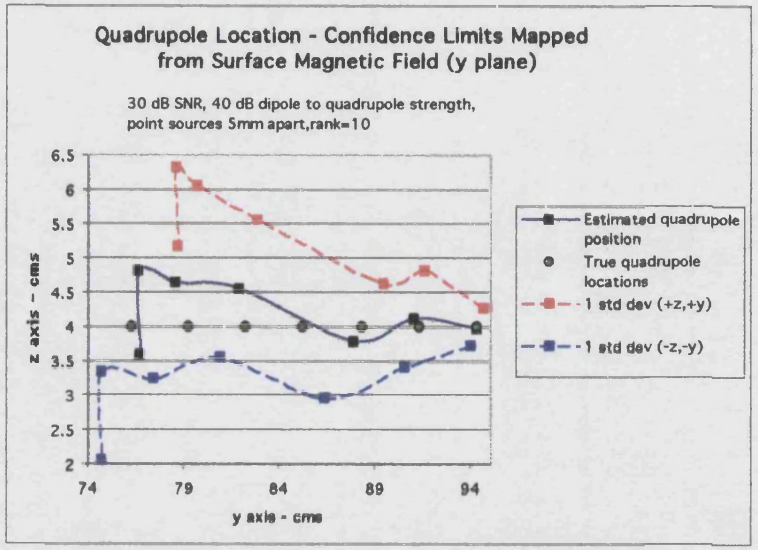


Fig 7b-5

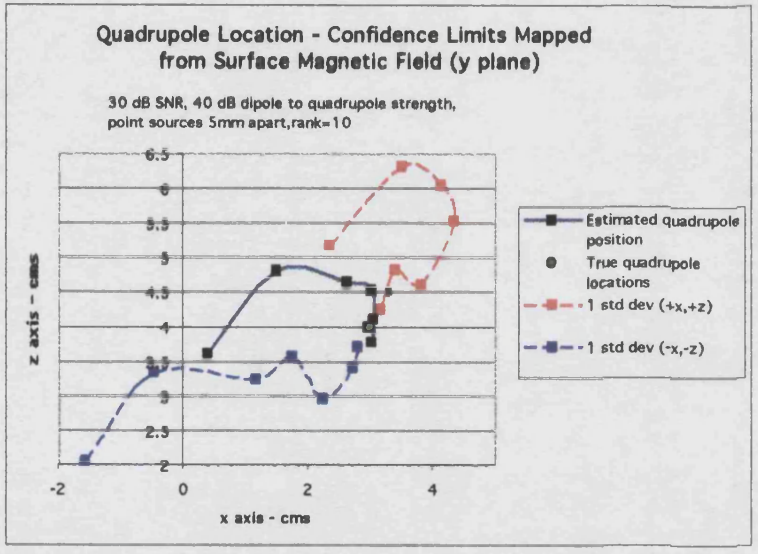


Fig 7b-6

Figs 7b-10 to 7b-12
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the quadrupole point current source using the surface magnetic field By. Rank deficient case rank=10.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.

Sensitivity of the Electric Potential and the Magnetic Field B_y Inverse to Source Depth – Rank=10

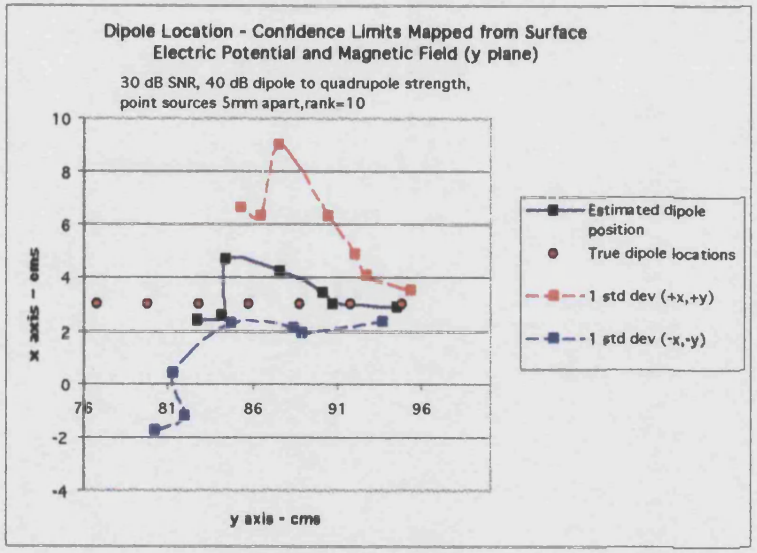


Fig 7b-7

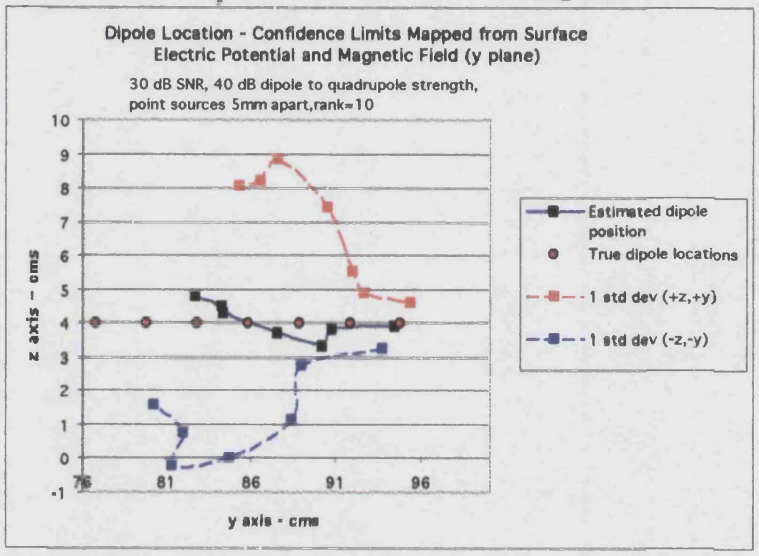


Fig 7b-8

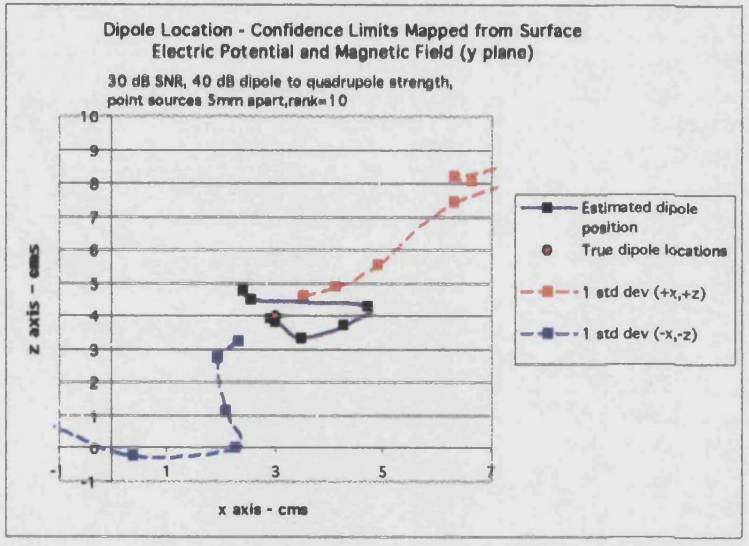


Fig 7b-9

Figs 7b-7 to 7b-9
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the dipole point current source using the surface electric potential and magnetic field By.
Rank deficient case rank=10.
Dipole at starts (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets.

Sensitivity of the Electric Potential and the Magnetic Field B_y Inverse to Source Depth – Rank=10

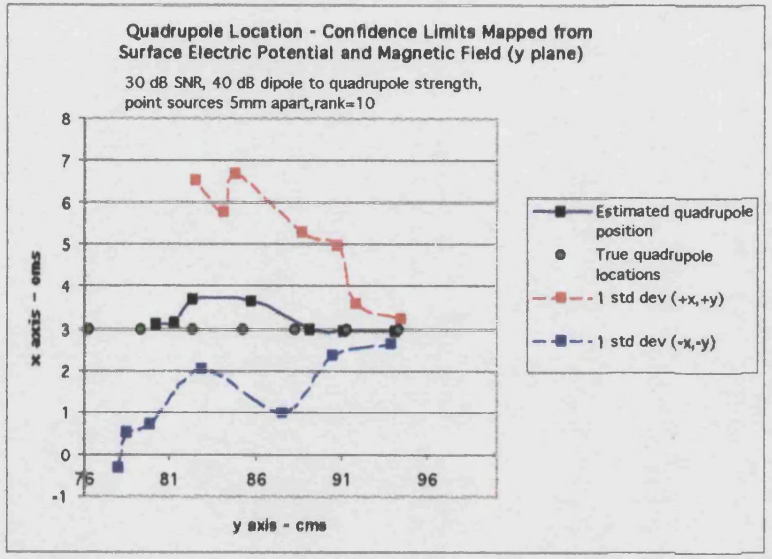


Fig 7b-10

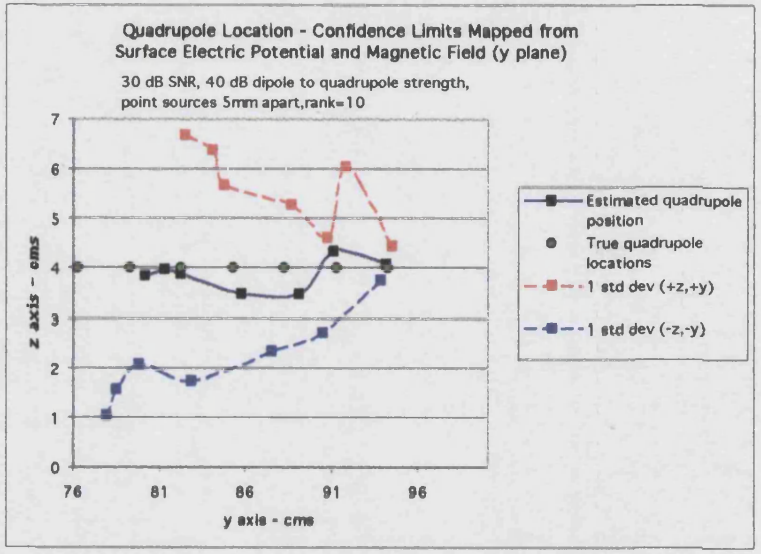


Fig 7b-11

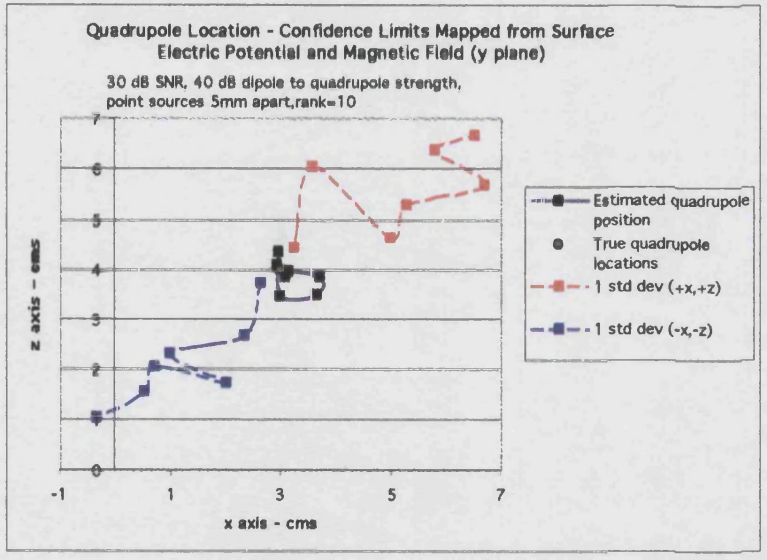


Fig 7b-12

Figs 7b-10 to 7b-12
x vs y, z vs y and z vs x plots –Depth Sensitivity
One standard deviation in the combined quadrupole and dipole inverse for the location of the quadrupole point current source using the surface electric potential and magnetic field By.
Rank deficient case rank=10.
Dipole starts at (3,94.8,4) and
Quadrupole starts at (3,94.3,4)
40 dB Dipole to Quadrupole relative strength
Noise at 30 dB SNR
Standard deviation from 40 to 50 data sets

Appendix 8 - Source Coding

The source coding shown here is one program from a suite of programs which are similar. This program is the full inverse model for separate quadrupole and dipole locations and allows any selection of measured data to be used. The number of measurement points for any data type (electric potential or any of the three magnetic planes) can be different in both number and surface positions (or the same).

The other programs, which are not presented, are designed for variations such as:

Dipole and quadrupole occupying the same position – inverse model

Dipole source only – inverse model

Magnetic inverse models with only the infinite medium field

Electric/magnetic only inverse models

Forward models for data generation

The reason for including the source coding is that the equations developed in Chapter 3 are not presented in full as they fairly lengthy and somewhat repetitive.

A brief description is first given of each function and what functions each calls in. The utility functions are also briefly described.

The function names follow a convention but as is normal during development departures from the convention occur and the sequencing gets out of step.

Program: DQHmcge2.π

Function name Model Functions	Purpose	Model Functions Called In
main	Sets up global variables, control parameters and starting assumptions Reads in data from files Computes field and other constants Sets inverse stopping parameters Stores results	field1a field2a DQV2magproj
DQV2magproj	Prepares data for calls to individual inverses Consolidates data following inverses Electric and magnetic inverse unification Error computations	HHdq2project HHdqmag3proj
field1a filed2a	Computes field measurement point constants for $m = m' + 1$, $m = m'$, $m = m' - 1$ These are the only valid integral equation solutions	
HHdq2project	The electric potential Variable Projection inverse: Is in 2 parts: linear variables Non-linear variables	fquad4ecg
HHdqmag3proj	The magnetic field Variable Projection inverse: Is in 2 parts: linear variables Non-linear variables	fharmmcg3d fharmmcg3b fdipmcg fquad1mcg fdip0mcg fquad0mcg
fquad4ecg	Computes the non-linear functionals for the electric dipole and quadrupole coefficients. Also computes the first differential of the same functionals.	Fxxx, Fxxy,Fxyz, Fxyx,Fxyy

fharmmcg3d	The infinite series computation of the surface magnetic field due to the volume currents from the dipole source. The function computes the non-linear functionals and also the first differentials. The truncation point is set in 'main'	
fharmmcg3b	The infinite series computation of the surface magnetic field due to the volume currents from the quadrupole source. The function computes the non-linear functionals and also the first differentials. The truncation point is set in 'main'	octapolea octapoleb makeFrr
fdipmcg	The infinite medium non-linear functionals for the dipole	
fquad1mcg	The infinite medium non-linear functionals for the quadrupole	
fdip0mcg	The first differential of the infinite medium non-linear functionals for the dipole	
fquad0mcg	The first differential of the infinite medium non-linear functionals for the quadrupole	Fxxx, Fxxy,Fxyz, Fxyx,Fxyy
Fxxx,Fxxy,Fxyz, Fxyx,Fxyy	Five small functions giving the third differential coefficient for the closed form lead field. These five cover all 27 variations for (x,y,z)	

Function name Utlity Functions	Purpose	Model Functions Called In
make2darray to make5darray	Reserve memory for array sizes specified in arguments	
free2darray to free5darray	Releases memory for array sizes specified in arguments	
ecgdata mcgdata	Reads in file data from specified text format files	
xyztosph sphtoxyz	Cartesian to spherical conversion Spherical to Cartesian conversion	
lgnd	Returns Associated Legendre coefficients for specified indices	
househ	Householder decompositon. Controlled by a permutation vector which contains the sequence in which column vectors passed to it are to be treated. Rank is a variable passed to the function.	
gaussj2	triangular matrix inversion	

The Model Functions

/**** DQHmcge2**

Separate locations for dipole and quadrupole

Inverse model which finds both the strength and location of a combined dipole and quadrupole source.
Model combines electric and magnetic field data. Field points can be different
for each and also for each magnetic field orthogonal plane. Data from any or all of the magnetic planes may be absent. Where
only one magnetic plane is present note the option switches which are available for the source strengths. These are
included because the linear dependencies in the equations mean that some of the source vectors are indeterminate
from the magnetic data and optionally those from the electric field may be used.

Dipole and quadrupole strength estimates are not combined. The best fit estimates for each magnetic/electric plane
are optionally kept separate for each data set. This model allows magnetic fields from
the volume currents at the boundary to be included. The dipole set is included in the solution (ie is part of the solution space)
The quadrupole set is estimated using the harmonic series
forward model which is input with location data and quadrupole strengths from the infinite medium
inverse model. The estimate is then subtracted from the measured data to provide a new estimate
for the infinite medium field which in turn is used in the next iteration of the infinite medium inverse model. This approach is used
as the solution was found to converge (unlike the dipole case). It could also be included in solution space if convergence is not
found (but not in this model)

Source location estimates are combined as follows:

- for the magnetic field data, quadrupole strengths are estimated for
each orthogonal plane and averaged taking account of the linear depenence in the equations.
Source location coordinates are then estimated for each orthogonal plane
and then averaged.

- for the electric field data, the 5 independent quadupole strengths are estimated
folllowed by the source location coordinates.

- the source location coordinates from the above are then averaged and the next
iteration is carried out depending on the total error.

The model reads in the number of data points followed by the surface xyz coordinates
and cooresponding measurements from chosen files. First the true source position
and quadrupole strengths are read in.
This inverse model uses Householder decompositons and the orthogonal projection method
to find the best fit dipole/quadrupole strengths and location from a least squares minimisation.
*******/

```
#include <stdio.h>
#include <stdlib.h>
#include <console.h>
#include <ctype.h>
#include <string.h>
#include <math.h>
#include "nrutil.h"
#include "in_data.h"

void field1a (int,float **,float **,float **,float ,float ,float *,float *,int,int );
void field2a (int,float **,float **,float **,float ,float ,float *,float *,int,int );
void DQV2magproj(FILE *,int,int,int,float *,float *,float *,float *,float *,int,int,float **,float **,float **,float **,float **,int,float ***,
                 float ***,float **,int,float ***,float ***,float **,int,float ***,float ***,float **,int,float *,float *);

main()
{

        float       *phi,*phinew,**Bx,**By,**Bz,err[ ]={0,0},errold[ ]={0,0},*da,*dae,*daq,*dad,*Cthex,*Sthex,**Cmphex,**Smphex,*Cthey
                    ,*Sthey,**Cmphey,**Smphey,*Cthez,*Sthez,**Cmphez,**Smphez,**Pe,**gma,**bc,ct,*a,*ae,*anew,astart[6],**xmx,**xmy,
                    **xmz,**xe,xsph[3],precision=0.001,vj[4],vjm[6],vje[6],radius,***BxB1,***ByB1,***BzB1,***BxA1,***ByA1,***BzA1,
                    ***BxB2,***ByB2,***BzB2,***BxA2,***ByA2,***BzA2;
        int         n,m,mrx=0,mry=0,mrz=0,mre=0,mremat,mrxmat,mrymat,mrzmat,ncmat,count=0,pcount=0,start,end,mflag=0,ind=0,fp1=1,fp2=1,fp3=1,
                        fp4=1,ctr[6],mloop;
        register int i,j,k;
        FILE        *ifp1,*ifp2,*ifp3,*ifp4,*ifp5,*ifp10,*ifp20,*ifp30,*ifp40,*ifp50;

        /*      Variables are:
                        3 X mr(xyz) arrays of m(xyz) (ie x,y,z) data points
                                Bx,By,Bz [0][m]     -       measured/true surface mag field
                                Bx,By,Bz [1][m]     -       estimated surface mag for inf medium
```

```
                    Bx,By,Bz [2][m]     -       estimated surface mag from volume currents

                3 element vectors
                    a(e)                -       source coordinates used to generate the data
                    aest(e,x,y,z)       -       estimated increment of source coordinates
                    anew(e,x,y,z)       -       new quadrupole increment of source coordinates for next iteration
                                        -       4sets e=electric x=Bx y=By z=Bz

                nc element vector
                    da(e)               -       magnetic (electric) quadrupole vector strengths used to generate the data
                    daest(e,x,y,z)      -       estimated quadrupole vector strengths
                    danew(e,x,y,z)      -       new quadrupole vector strengths for next iteration
                                        -       4sets e=electric x=Bx y=By z=Bz

                mr(xyz) x nc matrices (in the quadrupole case 'rank'= 5 = nc non-linear functionals)
                    xm(xyz) or (e)      -       mr surface coordinates

        */

/***** Enter input and output filenames here ***/

        char    fname1[ ]="::dp2Cndata:DQadp4Cl1x0",fname2[ ]="::dp2Cndata:DQadp4Cl1y0",fname3[ ]="::dp2Cndata:DQadp4Cl1z0",
                fname4[ ]="::dp2Cndata:DQadp4Cl1e0",ans='n';

        //For a single statistical noise run
        char    fname5[ ]="::dp2Cndata:results:ResDQadp4Cl1x0";
        char    fname10[ ]="::dpCndata:DQadpChx0",fname20[ ]="::dpCndata:DQadpChy0",fname30[ ]="::dpCndata:DQadpChz0",
fname40[ ]="::dpCndata:DQadpChe0";


        //For a series statistical noise run
        char    fnme[7][5][40];

        //Series
        strcpy (fnme[0][0], "::dp2Cndata:DQadp0Cl1x");strcpy (fnme[0][1], "::dp2Cndata:DQadp0Cl1y");
        strcpy (fnme[0][2], "::dp2Cndata:DQadp0Cl1z");strcpy (fnme[0][3], "::dp2Cndata:DQadp0Cl1e");
        strcpy (fnme[0][4], "::dp2Cndata:results:ResDQadp0Cl1yeNo");
        strcpy (fnme[1][0], "::dp2Cndata:DQadp1Cl1x");strcpy (fnme[1][1], "::dp2Cndata:DQadp1Cl1y");
        strcpy (fnme[1][2], "::dp2Cndata:DQadp1Cl1z");strcpy (fnme[1][3], "::dp2Cndata:DQadp1Cl1e");
        strcpy (fnme[1][4], "::dp2Cndata:results:ResDQadp1Cl1yeNo");
        strcpy (fnme[2][0], "::dp2Cndata:DQadp2Cl1x");strcpy (fnme[2][1], "::dp2Cndata:DQadp2Cl1y");
        strcpy (fnme[2][2], "::dp2Cndata:DQadp2Cl1z");strcpy (fnme[2][3], "::dp2Cndata:DQadp2Cl1e");
        strcpy (fnme[2][4], "::dp2Cndata:results:ResDQadp2Cl1yeNo");
        strcpy (fnme[3][0], "::dp2Cndata:DQadp3Cl1x");strcpy (fnme[3][1], "::dp2Cndata:DQadp3Cl1y");
        strcpy (fnme[3][2], "::dp2Cndata:DQadp3Cl1z");strcpy (fnme[3][3], "::dp2Cndata:DQadp3Cl1e");
        strcpy (fnme[3][4], "::dp2Cndata:results:ResDQadp3Cl1yeNo");
        strcpy (fnme[4][0], "::dp2Cndata:DQadp4Cl1x");strcpy (fnme[4][1], "::dp2Cndata:DQadp4Cl1y");
        strcpy (fnme[4][2], "::dp2Cndata:DQadp4Cl1z");strcpy (fnme[4][3], "::dp2Cndata:DQadp4Cl1e");
        strcpy (fnme[4][4], "::dp2Cndata:results:ResDQadp4Cl1yeNo");
        strcpy (fnme[5][0], "::dp2Cndata:DQadp5Cl1x");strcpy (fnme[5][1], "::dp2Cndata:DQadp5Cl1y");
        strcpy (fnme[5][2], "::dp2Cndata:DQadp5Cl1z");strcpy (fnme[5][3], "::dp2Cndata:DQadp5Cl1e");
        strcpy (fnme[5][4], "::dp2Cndata:results:ResDQadp5Cl1yeNo");
        strcpy (fnme[6][0], "::dp2Cndata:DQadp6Cl1x");strcpy (fnme[6][1], "::dp2Cndata:DQadp6Cl1y");
        strcpy (fnme[6][2], "::dp2Cndata:DQadp6Cl1z");strcpy (fnme[6][3], "::dp2Cndata:DQadp6Cl1e");
        strcpy (fnme[6][4], "::dp2Cndata:results:ResDQadp6Cl1yeNo");

        //Add series marker

        //Series 1
/*      for (i=0;i<7;i++) {
                for (j=0;j<5;j++) strcat(fnme[i][j],"1");
        }
*/

        //Series 1a
/*      for (i=0;i<7;i++) {
                strcat(fnme[i][4],"a");
        }
*/

        //Series 2
        for (i=0;i<7;i++) {
                for (j=0;j<5;j++) strcat(fnme[i][j],"2");
        }


        //Series 2a
/*      for (i=0;i<7;i++) {
```

```
                    strcat(fnme[i][4],"a");
            }
*/

            /**** Set to zero any field measurement plane not being used ****/
            fp1=fp3=0;  //<<<---- change these ---<<---
            //**** and ALSO inside inverse loop ******



            //*** open surface coordinate data input files for magnetic field data****

            //Set all file pointers to null
            ifp1=ifp2=ifp3=ifp4=NULL;

            //Set series run file pointers to zero
            ifp10=ifp20=ifp30=ifp40=ifp50=NULL;

/*****************************/



            if (fp1 != 0)
            {
                    mflag=1;
                    if((ifp1=fopen(fname1,"r")) == NULL) {
                                    printf("Cannot open file %s\n",fname1);
                                    return;
                    }

/******For a single statistical noise run
                    if((ifp10=fopen(fname10,"r")) == NULL) {
                                    printf("Cannot open file %s\n",fname10);
                                    return;
                    }
*****/

            }

            if (fp2 != 0)
            {

                    mflag=1;
                    if((ifp2=fopen(fname2,"r")) == NULL) {
                                    printf("Cannot open file %s\n",fname2);
                                    return;
                    }

/***** For a single statistical noise run
                    if((ifp20=fopen(fname20,"r")) == NULL) {
                                    printf("Cannot open file %s\n",fname20);
                                    return;
                    }
******/

            }

            if (fp3 != 0)
            {
                    mflag=1;
                    if((ifp3=fopen(fname3,"r")) == NULL) {
                                    printf("Cannot open file %s\n",fname3);
                                    return;
                    }

/***** For a single statistical noise run
                    if((ifp30=fopen(fname30,"r")) == NULL) {
                                    printf("Cannot open file %s\n",fname30);
                                    return;
                    }
******/

            }

            //*** open surface coordinate data input file for electric field data****
            if (fp4 != 0)
```

```
        {
                if((ifp4=fopen(fname4,"r")) == NULL) {
                                printf("Cannot open file %s\n",fname4);
                                return;
                }

/***** For a single statistical noise run
                if((ifp40=fopen(fname40,"r")) == NULL) {
                                printf("Cannot open file %s\n",fname40);
                                return;
                }
*****/

        }

        //*** open output file for surface coordinate data and computed potentials ****
/***** For a single statistical noise run******/
        if((ifp5=fopen(fname5,"a+")) == NULL) {
                                printf("Cannot open file %s\n",fname5);
                                return;
                                }


/*****************************/


/**** Set the multipole order ******/
        start=1;
        end=7;

        gma=make2darray(&end,&end);
        bc=make2darray(&end,&end);

        for (n=0;n<=end;n++){
                for (m=0;m<=n;m++){
                        gma[n][m]=(2-(m==0))*(fact(n-m))/(fact(n+m));
                 }
        }

        for (n=1;n<=end;n++){
                ct=((2*n+1)/(float)n);
                for (m=0;m<=n;m++){
                        bc[n][m] = ct*gma[n][m];
                }
         }
/***********************************/

/******** read in surface x, y and z coordinates ******/

        //getfilename(fname1); Use this for runtime file selection

        //[0],[1] and [2] are dipole [3],[4] and [5] are quadrupole
        ac=malloc(6*sizeof(float));
        a=malloc(6*sizeof(float));
        anew=malloc(6*sizeof(float));
        dad=malloc(12*sizeof(float));


        //Dipole strengths are at positions [9],[10] and [11]
        dac=malloc(12*sizeof(float));
        daq=malloc(12*sizeof(float));
        da=malloc(12*sizeof(float));




/*** Magnetic field data ***/


        //Data for x

        if (ifp1 != NULL)
        {
                ind +=1;

                fscanf(ifp1,"%f %f %f %f %f %f\n",a,a+1,a+2,a+3,a+4,a+5);
                fscanf(ifp1,"%f %f %f %f %f %f %f %f %f %f %f %f\n",daq,daq+1,daq+2,daq+3,daq+4,daq+5,daq+6,daq+7,daq+9,daq+10,daq+11);
```

```
		fscanf(ifp1,"%d\n",&mrx);


		mrxmat=mrx-1;
		ncmat=2;

		xmx=make2darray(&mrxmat,&ncmat);
		Bx=make2darray(&ncmat,&mrxmat);



		//Assume that Qxx+Qyy+Qzz=0
		daq[8]=-(daq[0]+daq[4]);

		ncmat=5;



		// read in mr coordinates into xmx
		//and mrx magnetic field measurements into Bx
		mcgdata2(ifp1,xmx,Bx[0],&mrx);

		for(i=0;i<mrx;i++)
		{
			Bx[1][i]=Bx[0][i];
		}


		fclose(ifp1);
}

//Data for y
if (ifp2 != NULL)
{
		ind +=3;

		fscanf(ifp2,"%f %f %f %f %f %f\n",a,a+1,a+2,a+3,a+4,a+5);
		fscanf(ifp2,"%f %f %f %f %f %f %f %f %f %f %f %f\n",daq,daq+1,daq+2,daq+3,daq+4,daq+5,daq+6,daq+7,daq+9,daq+10,daq+11);
		fscanf(ifp2,"%d\n",&mry);


		mrymat=mry-1;
		ncmat=2;

		xmy=make2darray(&mrymat,&ncmat);
		By=make2darray(&ncmat,&mrymat);

		//Assume that Qxx+Qyy+Qzz=0
		daq[8]=-(daq[0]+daq[4]);

		ncmat=5;



		// read in mr coordinates into xmy
		//and mry magnetic field measurements into By
		mcgdata2(ifp2,xmy,By[0],&mry);
		for(i=0;i<mry;i++)
		{
			By[1][i]=By[0][i];
		}


		fclose(ifp2);
}

//Data for z
if (ifp3 != NULL)
{
		ind +=5;

		fscanf(ifp3,"%f %f %f %f %f %f\n",a,a+1,a+2,a+3,a+4,a+5);
		fscanf(ifp3,"%f %f %f %f %f %f %f %f %f %f %f %f\n",daq,daq+1,daq+2,daq+3,daq+4,daq+5,daq+6,daq+7,daq+9,daq+10,daq+11);
		fscanf(ifp3,"%d\n",&mrz);


		mrzmat=mrz-1;
		ncmat=2;
```

```
                xmz=make2darray(&mrzmat,&ncmat);
                Bz=make2darray(&ncmat,&mrzmat);

                //Assume that Qxx+Qyy+Qzz=0
                daq[8]=-(daq[0]+daq[4]);

                ncmat=5;

                // read in mr coordinates into xmz
                //and mrz magnetic field measurements into Bz
                mcgdata2(ifp3,xmz,Bz[0],&mrz);
                for(i=0;i<mrz;i++)
                {
                        Bz[1][i]=Bz[0][i];
                }


                fclose(ifp3);
        }

/*** Electric field data ***/

        if (ifp4 != NULL)
        {
                fscanf(ifp4,"%f %f %f %f %f %f\n",ae,ae+1,ae+2,ae+3,ae+4,ae+5);
                fscanf(ifp4,"%f %f %f %f %f %f %f %f\n",dae,dae+1,dae+2,dae+3,dae+4,dae+5,dae+6,dae+7);
                fscanf(ifp4,"%d\n",&mre);

                mremat=mre-1;
                ncmat=2;

                xe=make2darray(&mremat,&ncmat);
                phi=malloc((mre)*sizeof(float));
                phinew=malloc((mre)*sizeof(float));

                ecgdata(ifp4,xe,phi,&mre); // reads mr records into **A
                printf("\n%15.9f\n",100*xe[0][1]);
                fclose(ifp4);
        }

/****************** End of data input ***************************/


/**** Set initial guess of source location ****/
        //[0],[1] and [2] are quadrupole
        anew[0]=0.06;anew[1]=0.85;anew[2]=0.02;

        //[3],[4] and [5] are dipole
        anew[3]=.02;anew[4]=0.91;anew[5]=0.06;

        //For unknown quadrupole strengths set
        //all values to 1. The normal assumption for unknown data.
        for (i=0;i<12;i++)          {da[i]=1;dae[i]=1;}

        //da[4] set to 0 so that QVproject initial guess is not distorted
        da[4]=0;

        //dae[0] and [3] set so that correction coding in QVproject makes them =1

        if (ifp4!=NULL) {
                dae[3]=dae[0]=(1.0/3);
        }
        else for (i=0;i<12;i++)     dae[i]=0;


        radius=pow((anew[0]*anew[0]+anew[1]*anew[1]+anew[2]*anew[2]),0.5);
        printf("Starting radius for quadrupole is %f\n\n",radius);
        if (radius>=1) {printf("\nStarting point is outside or on surface");return 0;}

        radius=pow((anew[3]*anew[3]+anew[4]*anew[4]+anew[5]*anew[5]),0.5);
        printf("Starting radius for dipole is %f\n\n",radius);
        if (radius>=1) {printf("\nStarting point is outside or on surface");return 0;}


/*** Set decomposition control for Householder calculation ***/
        //For magnetic projection
          vj[0]=10000;
```

```
 vj[1]=10000;
 vj[2]=10000;
//For electric projection
 vj[3]=100000;


/**** Compute 'source independent' field parameters *****/

        //convert surface coords to spherical
        //and fill in sin and cos arrays

        ncmat=end+3;
        Pe=make2darray(&ncmat,&ncmat);

        //For x data
        if (ifp1 != NULL)
        {
                Sthex=malloc((mrx)*sizeof(float));
                Cthex=malloc((mrx)*sizeof(float));
                Smphex=make2darray(&mrxmat,&ncmat);
                Cmphex=make2darray(&mrxmat,&ncmat);

                for (i=0;i<mrx;i++) {

                        //Convert to spherical
                        xyztosph(xmx[i],xsph);

                        Sthex[i]=sin(xsph[1]);
                        Cthex[i]=cos(xsph[1]);

                        for (j=0;j<=(end+2);j++) {
                          Smphex[i][j]=sin(j*xsph[2]);
                          Cmphex[i][j]=cos(j*xsph[2]);
                        }

                }

        /* The surface data points are evaluated once from the nabla cross product and become
          constants in the inverse solution.
          These values are temporarily held as even and odd values in BxB .... BzA for
          each data point and each n and m of the harmonic series.  They are recombined
          with the even and odd source terms at each iteration */

                BxB1=make3darray(&mrxmat,&end,&end);
                BxA1=make3darray(&mrxmat,&end,&end);
                BxB2=make3darray(&mrxmat,&end,&end);
                BxA2=make3darray(&mrxmat,&end,&end);

                for (i=0;i<mrx;i++) {

                        //Loop for each data point
                        lgnd(Pe,Cthex[i],ncmat);
                        field1a(0,BxB1[i],BxA1[i],Pe,Cthex[i],Sthex[i],Cmphex[i],Smphex[i],start,end);
                        field2a(0,BxB2[i],BxA2[i],Pe,Cthex[i],Sthex[i],Cmphex[i],Smphex[i],start,end);

                        //Add the (m'+1) and (m'-1) series
                        for (n=1;n<=end;n++){
                        for (m=0;m<=n;m++){
                                BxA1[i][n][m] +=BxA2[i][n][m];
                                        BxB1[i][n][m] +=BxB2[i][n][m];
                        }
                   }
                }
        }

        //For y data
        if (ifp2 != NULL)
        {
                Sthey=malloc((mry)*sizeof(float));
                Cthey=malloc((mry)*sizeof(float));
                Smphey=make2darray(&mrymat,&ncmat);
                Cmphey=make2darray(&mrymat,&ncmat);

                for (i=0;i<mry;i++) {

                        //Convert to spherical
                        xyztosph(xmy[i],xsph);
```

```
                Sthey[i]=sin(xsph[1]);
                Cthey[i]=cos(xsph[1]);

                for (j=0;j<=(end+2);j++) {
                  Smphey[i][j]=sin(j*xsph[2]);
                  Cmphey[i][j]=cos(j*xsph[2]);
                }

        }

/* The surface data points are evaluated once from the nabla cross product and become
   constants in the inverse solution.
   These values are temporarily held as even and odd values in BxB .... BzA for
   each data point and each n and m of the harmonic series.  They are recombined
   with the even and odd source terms at each iteration */

        ByB1=make3darray(&mrymat,&end,&end);
        ByA1=make3darray(&mrymat,&end,&end);
        ByB2=make3darray(&mrymat,&end,&end);
        ByA2=make3darray(&mrymat,&end,&end);

        for (i=0;i<mry;i++) {

                //Loop for each data point
                lgnd(Pe,Cthey[i],ncmat);
                field1a(1,ByB1[i],ByA1[i],Pe,Cthey[i],Sthey[i],Cmphey[i],Smphey[i],start,end);
                field2a(1,ByB2[i],ByA2[i],Pe,Cthey[i],Sthey[i],Cmphey[i],Smphey[i],start,end);

                //Add the (m'+1) and (m'-1) series
                for (n=1;n<=end;n++){
                for (m=0;m<=n;m++){
                                        ByA1[i][n][m] +=ByA2[i][n][m];
                                        ByB1[i][n][m] +=ByB2[i][n][m];
                }
           }
        }
}

//For z data
if (ifp3 != NULL)
{
        Sthez=malloc((mrz)*sizeof(float));
        Cthez=malloc((mrz)*sizeof(float));
        Smphez=make2darray(&mrzmat,&ncmat);
        Cmphez=make2darray(&mrzmat,&ncmat);

        for (i=0;i<mrz;i++) {

                //Convert to spherical
                xyztosph(xmz[i],xsph);

                Sthez[i]=sin(xsph[1]);
                Cthez[i]=cos(xsph[1]);

                for (j=0;j<=(end+2);j++) {
                  Smphez[i][j]=sin(j*xsph[2]);
                  Cmphez[i][j]=cos(j*xsph[2]);
                }

        }

/* The surface data points are evaluated once from the nabla cross product and become
   constants in the inverse solution.
   These values are temporarily held as even and odd values in BxB .... BzA for
   each data point and each n and m of the harmonic series.  They are recombined
   with the even and odd source terms at each iteration */

        BzB1=make3darray(&mrzmat,&end,&end);
        BzA1=make3darray(&mrzmat,&end,&end);
        BzB2=make3darray(&mrzmat,&end,&end);
        BzA2=make3darray(&mrzmat,&end,&end);

        for (i=0;i<mrz;i++) {

                //Loop for each data point
                lgnd(Pe,Cthez[i],ncmat);
                field1a(2,BzB1[i],BzA1[i],Pe,Cthez[i],Sthez[i],Cmphez[i],Smphez[i],start,end);
                field2a(2,BzB2[i],BzA2[i],Pe,Cthez[i],Sthez[i],Cmphez[i],Smphez[i],start,end);
```

```
                        //Add the (m'+1) and (m'-1) series
                        for (n=1;n<=end;n++){
                        for (m=0;m<=n;m++){
                                        BzA1[i][n][m] +=BzA2[i][n][m];
                                        BzB1[i][n][m] +=BzB2[i][n][m];
                        }
                    }
                }
        }

/*********************************************************/

/***** Inverse solution **********/

/***** Outer loop to control series run ****/

for (mloop=0;mloop<7;mloop++)
{
                fp1=fp2=fp3=1;
                if(mloop==0) fp1=fp3=0;
                if(mloop==1) fp1=fp3=0;
                if(mloop==2) fp1=fp3=0;
                if(mloop==3) fp1=fp3=0;
                if(mloop==4) fp1=fp3=0;
                if(mloop==5) fp1=fp3=0;
                if(mloop==5) fp1=fp3=0;
                if(mloop==6) fp1=fp3=0;

                //*** open surface coordinate data input file for named field data****

                if (fp1 != 0) {
                        if((ifp10=fopen(fnme[mloop][0],"r")) == NULL) {
                                        printf("Cannot open file %s\n",fnme[mloop][0]);
                                        return;
                                }
                }

                if (fp2 != 0) {
                        if((ifp20=fopen(fnme[mloop][1],"r")) == NULL) {
                                        printf("Cannot open file %s\n",fnme[mloop][1]);
                                        return;
                                }
                }

                if (fp3 != 0) {
                        if((ifp30=fopen(fnme[mloop][2],"r")) == NULL) {
                                        printf("Cannot open file %s\n",fnme[mloop][2]);
                                        return;
                                }
                }

                if (ifp4 != NULL) {
                        if((ifp40=fopen(fnme[mloop][3],"r")) == NULL) {
                                        printf("Cannot open file %s\n",fnme[mloop][3]);
                                        return;
                                }
                }


        //*** open output file for surface coordinate data and computed potentials ****

        if((ifp50=fopen(fnme[mloop][4],"a+")) == NULL) {
                                        printf("Cannot open file %s\n",fnme[mloop][4]);
                                        return;
                                        }


        for (i=0;i<50;i++) {

        //For a series statistical noise run
                if (fp1 != 0) {for (j=0;j<mrx;j++) fscanf(ifp10,"%f ",&Bx[1][j]);}
                if (fp2 != 0) {for (j=0;j<mry;j++) fscanf(ifp20,"%f ",&By[1][j]);}
                if (fp3 != 0) {for (j=0;j<mrz;j++) fscanf(ifp30,"%f ",&Bz[1][j]);}
                if (ifp4 != NULL) {for (j=0;j<mre;j++) fscanf(ifp40,"%f ",&phi[j]);}
```

```
//if(i>29){

        //For unknown multipole strengths reset
        //all values to 1. The normal assumption for unknown data.
        for (j=0;j<12;j++)      {da[j]=1;dae[j]=1;}

        if (ifp4!=NULL) {
                //dae[3]=dae[0]=(1.0/3);
                dae[3]=dae[0]=1;
        }
        else for (j=0;j<12;j++)   dae[j]=0;

        /**** Reset initial guess of source location ****/

        //[3],[4] and [5] are DIPOLE (true=[0.03,0.948,0.4])
        anew[3]=astart[3]=0.02;anew[4]=astart[4]=0.86;anew[5]=astart[5]=0.02;

        //[0],[1] and [2] are QUADRUPOLE (true=[0.03,0.943,0.4])
        anew[0]=astart[0]=0.02;anew[1]=astart[1]=0.80;anew[2]=astart[2]=0.02;


        /*** Reset decomposition control for Householder calculation ***/
        //For magnetic projection
        vj[0]=vjm[0]=100;
        vj[1]=vjm[0]=100;
        vj[2]=vjm[0]=100;
        //For electric projection
        vj[3]=vje[0]=1000;

        ctr[0]=10;ctr[1]=20;ctr[2]=30;ctr[3]=40;ctr[4]=80;ctr[5]=250;
        count=0;
        pcount=0;


do      {
        count++;
        pcount++;
        errold[0]=err[0];errold[1]=err[1];

        if (count>ctr[0]) {vj[0]=vj[1]=vj[2]=vjm[0]=100;vj[3]=vje[0]=100;}
        if (count>ctr[1]) {vj[0]=vj[1]=vj[2]=vjm[1]=10;vj[3]=vje[1]=10;}
        if (count>ctr[2]) {vj[0]=vj[1]=vj[2]=vjm[2]=10;vj[3]=vje[2]=10;}
        if (count>ctr[3]) {vj[0]=vj[1]=vj[2]=vjm[3]=10;vj[3]=vje[3]=10;}
        if (count>ctr[4]) {vj[0]=vj[1]=vj[2]=vjm[4]=10;vj[3]=vje[4]=10;}
if (count>ctr[5]) {vj[0]=vj[1]=vj[2]=vjm[4]=1;vj[3]=vje[4]=500;}

        //Shallow global minimum
        //if (count==80) anew[4]=0.02+anew[1];

        //Deep global minimum
        //if (count==80) anew[1]=anew[4]-0.02;



        DQV2magproj(ifp4,ind,mflag,err,anew,da,vj,start,end,bc,Bx,By,Bz,xmx,mrx,BxA1,BxB1,xmy,mry,ByA1,ByB1,xmz,mrz,BzA1,
                BzB1,xe,mre,phi,dae);

        printf("\n\nElectric field projection:");
        printf("\nPhi error sqrd=%15.8f",err[1]);
        printf("\nNext guess Dx=%f  Dy=%f  Dz=%f  \nQxx=%f  Qxy=%f  Qxz=%f  Qyy=%f  Qyz=%f\n",dae[5],dae[6],dae[7],dae[0],dae[1],
                dae[2],dae[3],dae[4]);
        printf("\n\nMagnetic field projection:");
        printf("\nB error sqrd=%15.8f\n",err[0]);
        printf("\nDx=%6.4f  Dy=%6.4f  Dz=%6.4f \n\n",da[9],da[10],da[11]);
        printf("Qxx=%6.4f  Qyx=%6.4f  Qzx=%6.4f \n",da[0],da[1],da[2]);
        printf("Qxy=%6.4f  Qyy=%6.4f  Qzy=%6.4f \n",da[3],da[4],da[5]);
        printf("Qxz=%6.4f  Qyz=%6.4f  Qzz=%6.4f \n",da[6],da[7],da[8]);
        printf("\nNew  locn. av. for dipole x=%8.5f  y=%8.5f  z=%8.5f",anew[3],anew[4],anew[5]);
        printf("\nNew  locn. av. for quadrupole x=%8.5f  y=%8.5f  z=%8.5f",anew[0],anew[1],anew[2]);
        printf("\nSeries no %d Set no: %d Iteration count is %d\n\n",mloop+1,i+1,count);
        printf("\n");

/*      fprintf(ifp50," dipole ");
        for (j=3;j<6;j++) fprintf(ifp50," %6.3f ",astart[j]);
        fprintf(ifp50," quadrupole ");
        for (j=0;j<3;j++) fprintf(ifp50," %6.3f ",astart[j]);
        fprintf(ifp50," dipole %8.5f %8.5f %8.5f",anew[3],anew[4],anew[5]);
        fprintf(ifp50," quadrupole %8.5f %8.5f %8.5f",anew[0],anew[1],anew[2]);
```

```
                fprintf(ifp50," error= %10.5f %10.5f ",err[0],err[1]);
                if (count==2) fprintf(ifp50,"\n");
*/

        }while (((err[0]<3000000000000.0)&&(err[0]>0.00000001)&&(count<150))||((err[1]<300000000000.0)&&(err[1]>0.00000001)&&(count<150)));


        //For a single statistical noise run

/*              fprintf(ifp5,"\nset %d ",i);
                fprintf(ifp5," Phi error sqrd= %10.5f delta error= %10.5f ",err[1],err[1]-errold[1]);
                fprintf(ifp5," Dx= %f  Dy= %f  Dz= %f Qxx= %f  Qxy= %f  Qxz= %f  Qyy= %f  Qyz= %f ",dae[5],dae[6],dae[7],dae[0],dae[1],
                        dae[2],dae[3],dae[4]);
                fprintf(ifp5," B error sqrd= %15.8f delta error= %10.5f ",err[0],err[0]-errold[0]);
                fprintf(ifp5," Dx= %6.4f  Dy= %6.4f  Dz= %6.4f  ",da[9],da[10],da[11]);
                fprintf(ifp5,"Qxx= %6.4f  Qyx= %6.4f  Qzx= %6.4f  ",da[0],da[1],da[2]);
                fprintf(ifp5,"Qxy= %6.4f  Qyy= %6.4f  Qzy= %6.4f  ",da[3],da[4],da[5]);
                fprintf(ifp5,"Qxz= %6.4f  Qyz= %6.4f  Qzz= %6.4f  ",da[6],da[7],da[8]);
                fprintf(ifp5," New  locn. av. for dipole x= %8.5f  y= %8.5f  z= %8.5f",anew[3],anew[4],anew[5]);
                fprintf(ifp5," New  locn. av. for quadrupole x= %8.5f  y= %8.5f  z= %8.5f",anew[0],anew[1],anew[2]);
                fprintf(ifp5," Iteration count was %d  ",count);


                fprintf(ifp5,"\nRun parameters ");
                if (ifp1 != NULL){
                        fprintf(ifp5,"%20s ",fname10);
                }
                if (ifp2 != NULL){
                        fprintf(ifp5,"%20s ",fname20);
                }
                if (ifp3 != NULL){
                        fprintf(ifp5,"%20s ",fname30);
                }
                if (ifp4 != NULL){
                        fprintf(ifp5,"%20s ",fname40);
                }
                fprintf(ifp5,"start dipole ");
                for (i=3;i<6;i++) fprintf(ifp5," %6.3f ",astart[i]);
                fprintf(ifp5,"quadrupole ");
                for (i=0;i<3;i++) fprintf(ifp5," %6.3f ",astart[i]);
                for (i=0;i<6;i++) fprintf(ifp5,"ctr %d vjm= %6.0f vje= %6.0f ",ctr[i],vjm[i],vje[i]);
*/

        //For a series statistical noise run

                fprintf(ifp50,"\nset %d ",i);
                fprintf(ifp50," Phi error sqrd= %10.5f delta error= %10.5f ",err[1],err[1]-errold[1]);
                fprintf(ifp50," Dx= %f  Dy= %f  Dz= %f Qxx= %f  Qxy= %f  Qxz= %f  Qyy= %f  Qyz= %f ",dae[5],dae[6],dae[7],dae[0],
                        dae[1],dae[2],dae[3],dae[4]);
                fprintf(ifp50," B error sqrd= %15.8f delta error= %10.5f ",err[0],err[0]-errold[0]);
                fprintf(ifp50," Dx= %6.4f  Dy= %6.4f  Dz= %6.4f  ",da[9],da[10],da[11]);
                fprintf(ifp50,"Qxx= %6.4f  Qyx= %6.4f  Qzx= %6.4f  ",da[0],da[1],da[2]);
                fprintf(ifp50,"Qxy= %6.4f  Qyy= %6.4f  Qzy= %6.4f  ",da[3],da[4],da[5]);
                fprintf(ifp50,"Qxz= %6.4f  Qyz= %6.4f  Qzz= %6.4f  ",da[6],da[7],da[8]);
                fprintf(ifp50," New  locn. av. for dipole x= %8.5f  y= %8.5f  z= %8.5f",anew[3],anew[4],anew[5]);
                fprintf(ifp50," New  locn. av. for quadrupole x= %8.5f  y= %8.5f  z= %8.5f",anew[0],anew[1],anew[2]);
                fprintf(ifp50," Iteration count was %d  ",count);

        //end of if i
        //}

        //End of i loop
        }
                fprintf(ifp50,"\nRun parameters ");
                if (fp1 != 0){
                        fprintf(ifp50,"%20s ",fnme[mloop][0]);
                }
                if (fp2 != 0){
                        fprintf(ifp50,"%20s ",fnme[mloop][1]);
                }
                if (fp3 != 0){
                        fprintf(ifp50,"%20s ",fnme[mloop][2]);
                }
                if (ifp4 != NULL){
                        fprintf(ifp50,"%20s ",fnme[mloop][3]);
                }
                fprintf(ifp50,"start dipole ");
                for (i=3;i<6;i++) fprintf(ifp50," %6.3f ",astart[i]);
```

```
                fprintf(ifp50,"quadrupole ");
                for (i=0;i<3;i++) fprintf(ifp50," %6.3f ",astart[i]);
                for (i=0;i<6;i++) fprintf(ifp50,"ctr %d vjm= %6.0f vje= %6.0f ",ctr[i],vjm[i],vje[i]);


        //For a single statistical noise run
        //fclose(ifp5);

        //For a series statistical noise run
                if (fp1 != 0){
                        fclose(ifp10);
                }
                if (fp2 != 0){
                        fclose(ifp20);
                }
                if (fp3 != 0){
                        fclose(ifp30);
                }
                if (ifp4 != NULL){
                        fclose(ifp40);
                }
        //End of mloop
        }

        return 0;
}
```

```
/***** DQV2magproj
        Calls specified inverses.
        Data is adjusted beforehand and consolidated afterwards  ****/


#include <stdio.h>
#include <stdlib.h>
#include <math.h>

void HHdqmag3proj (int,int,float **,int,float *,float *,float *,float *,float *,float *,float,int *,int *,float ***,float ***,int,int,float **);
void HHdq2project       (float **,int,int,int,float *,float *,float *,float *,float *,float *,float );


void DQV2magproj(FILE *ifp4,int ind,int mflag,float *err,float *a,float *da,float *vj,int start,int end,float ** bc,float **Bx,float
**By,float **Bz,float **xmx,int mrx,float ***BxA1,float ***BxB1,float **xmy,int mry,float
***ByA1,float ***ByB1,float **xmz,int mrz,float ***BzA1,float ***BzB1,float **xe,int mre,float *phi,float *dae)
{

        float       *Bxnew,*Bynew,*Bznew,*phinew,
                    *daeste,*daestx,*daesty,*daestz,*dav,
                    *anewe,*aeste,*anew,*anewacc,*anewold,*aestx,*aesty,*aestz,*ae,
                    Berrold,phierrold,dax0,day0,daz0,Dx[12],Dy[12],Dz[12];
        int         avge,nz,n,m,nc=6,rank=5,mremat,mrxmat,mrymat,mrzmat,ncmat,count=0,pcount=0,
qx[ ]={0,3,5,6,7,10,11},qxv[ ]={0,1,2,3,4,5,6,7,8,9,10,11},qy[ ]={0,1,2,6,7,9,11},
qyv[ ]={0,1,2,3,4,5,6,7,8,9,10,11},qz[ ]={0,1,2,3,5,9,10},qzv[ ]={0,1,2,3,4,5,6,7,8,9,10,11},option;
        register int  i,j;


        /*      The main control function for one iteration loop.
                Variable projection functions are called depending on which planes have been selected in
                main.
                Unification options can switched in or out.
                If rank deficient rune are made, then the use of qx[] to qzv[ ] can force column vectors to be excluded.



        Variables are:
                3 X mr(xyz) arrys of m(xyz) (ie x,y,z) data points
                        Bx,By,Bz [0][m]         -       measured/true surface mag field
                        Bx,By,Bz [1][m]         -       estimated surface mag field from the dipole and from the inf medium part of the quadrupole
                        Bx,By,Bz [2][m]         -       estimated surface mag from volume currents of the quadrupole

                6 element vectors       [0],[1] and [2] are dipole [3],[4] and [5] are quadrupole
                        a(e)                    -       source coordinates used to generate the data
                        aest(e,x,y,z)           -       estimated increment of source coordinates
                        anew(e,x,y,z)           -       new quadrupole increment of source coordinates for next iteration
                                                                -       4sets e=electric x=Bx y=By z=Bz

                nc element vector
                        da(e)                   -       magnetic (electric) dipole vector strengths used to generate the data
                        daest(e,x,y,z)          -       estimated dipole vector strengths
                        danew(e,x,y,z)          -       new dipole vector strengths for next iteration
                                                                -       4sets e=electric x=Bx y=By z=Bz

                mr(xyz) x nc matrices (in the dipole magnetic case 'rank'= 2 = nc non-linear functionals and
                                        in the dipole electric case 'rank'= 3 = nc non-linear functionals))
                        xm(xyz) or (e)          -       mr surface coordinates

        */


        ae=malloc(6*sizeof(float));
        anewe=malloc(6*sizeof(float));
        aeste=malloc(6*sizeof(float));
        aestx=malloc(6*sizeof(float));
        anew=malloc(6*sizeof(float));
        aesty=malloc(6*sizeof(float));
        aestz=malloc(6*sizeof(float));
        anewacc=malloc(6*sizeof(float));
        anewold=malloc(6*sizeof(float));

        //Dipole strengths are at positions [9],[10] and [11]
        dav=malloc(12*sizeof(float));
        daeste=malloc(12*sizeof(float));
        daestx=malloc(12*sizeof(float));
        daesty=malloc(12*sizeof(float));
        daestz=malloc(12*sizeof(float));
```

```
phinew=malloc(mre*sizeof(float));
Bxnew=malloc(mrx*sizeof(float));
Bynew=malloc(mry*sizeof(float));
Bznew=malloc(mrz*sizeof(float));

for (i=0;i<12;i++) {
        daestx[i]=daesty[i]=daestz[i]=daeste[i]=0;
}
for (i=0;i<6;i++) {
aestx[i]=aesty[i]=aestz[i]=aeste[i]=0;
}


        //Compute the electric field source coordinates


        if (ifp4 != NULL)
        {
                nc=8; rank=8;

                //Quadrupole diagonal is reset for HHdqproject (temp use of dae[8])
                //Note that dae is ordered as:  Qxx Qxy Qxz Qyy Qyz  for elements [0,1,2,3,4] and Dx Dy and Dz [5, 6, 7]

                dae[8]=2*dae[0]+dae[3];
                dae[3]=2*dae[3]+dae[0];
                dae[1]=dae[1]*2;
                dae[2]=dae[2]*2;
                dae[4]=dae[4]*2;
                dae[0]=dae[8];


                //HHdq reverses dipole and quadrupole coordinates
                for (i=0;i<3;i++){
                        ae[i]=a[i+3];
                        ae[i+3]=a[i];
                }


                HHdq2project(xe,mre,nc,rank,ae,aeste,dae,daeste,phi,phinew,vj[3]);

                //Swap back coordinates and test for >1
                for (i=0;i<3;i++) {
                        anewe[i+3] = ae[i] - aeste[i];
                        anewe[i] = ae[i+3] - aeste[i+3];
                        if (anewe[i]>1) anewe[i]=1;
                        if (anewe[i]<-1) anewe[i]=-1;
                        if (anewe[i+3]>1) anewe[i+3]=1;
                        if (anewe[i+3]<-1) anewe[i+3]=-1;
                }

                phierrold=err[1];
                err[1]=0;

                for (i=0;i<mre;i++) {
                        err[1] += pow((phi[i]-phinew[i]),2);
                }

        }



        //Using the new source estimate and best source location estimate, compute  correction
        //for the magnetic field

                for (i=0;i<6;i++) {
                        anewacc[i] = 0;
                }

                Berrold=err[0];
                err[0]=0;
                avge=0;

                for (i=0;i<12;i++){
                        Dx[i]=Dy[i]=Dz[i]=0;
                }
```

```
		for (i=0;i<12;i++){
			Dx[i]= da[i];
			Dy[i]= da[i];
			Dz[i]= da[i];

		}

		rank=12;
		if ((ind==1)||(ind==4)||(ind==6)||(ind==9))
		{

			HHdqmag3proj(0,ind,xmx,mrx,a,aestx,Dx,daestx,Bx[1],Bxnew,vj[0],qx,qxv,BxA1,BxB1,start,end,bc);
			for (i=0;i<6;i++) anewacc[i] += a[i] - aestx[i];
			avge++;

		}


		if ((ind==3)||(ind==4)||(ind==8)||(ind==9))
		{
			HHdqmag3proj(1,ind,xmy,mry,a,aesty,Dy,daesty,By[1],Bynew,vj[1],qy,qyv,ByA1,ByB1,start,end,bc);
			for (i=0;i<6;i++) anewacc[i] += a[i] - aesty[i];
			avge++;
		}


		if ((ind==5)||(ind==6)||(ind==8)||(ind==9))
		{
			HHdqmag3proj(2,ind,xmz,mrz,a,aestz,Dz,daestz,Bz[1],Bznew,vj[2],qz,qzv,BzA1,BzB1,start,end,bc);
			for (i=0;i<6;i++) anewacc[i] += a[i] - aestz[i];
			avge++;
		}


//Compute average new location from magnetic field inverse
if (mflag) {
		for (i=0;i<6;i++) {
			a[i] = anewacc[i]/avge;
			if (a[i]>1) a[i]=1;
			if (a[i]<-1) a[i]=-1;
			anewold[i]=a[i];
		}
}


//Compute here estimate of complete quadrupole vector da[i].
for (i=0;i<12;i++) da[i]=0;

if    (ind==1) for (i=0;i<rank;i++) da[qxv[i]]=daestx[qxv[i]];
else if (ind==3) for (i=0;i<rank;i++) da[qyv[i]]=daesty[qyv[i]];
else if (ind==5) for (i=0;i<rank;i++) da[qzv[i]]=daestz[qzv[i]];
else {
		if (ind==4)
		{
			da[0]=(4*daesty[0]-daestx[0])/3;
			da[1]=daesty[1];
			da[2]=daesty[2];
			da[3]=daestx[3];
			da[4]=(daestx[0]-daesty[0])*2/3;
			da[5]=daestx[5];
			da[6]=(daestx[6]+daesty[6])/2;
			da[7]=(daestx[7]+daesty[7])/2;
			da[8]=-(da[0]+da[4]);
			da[9]=daesty[9];
			da[10]=daestx[10];
			da[11]=(daestx[11]+daesty[11])/2;
		}
		else if (ind==6)
		{
			da[0]=(daestx[0]+2*daestz[0])/3;
			da[1]=daestz[1];
			da[2]=daestz[2];
			da[3]=(daestx[3]+daestz[3])/2;
			da[4]=(daestx[0]-daestz[0])/3;
			da[5]=(daestx[5]+daestz[5])/2;
			da[6]=daestx[6];
			da[7]=daestx[7];
```

```
                da[8]=-(da[0]+da[4]);
                da[9]=daestz[9];
                da[10]=(daestx[10]+daestz[10])/2;
                da[11]=daestx[11];

        }
        else if (ind==8)
        {
                da[0]=(2*daesty[0]+daestz[0])/3;
                da[1]=(daesty[1]+daestz[1])/2;
                da[2]=(daesty[2]+daestz[2])/2;
                da[3]=daestz[3];
                da[4]=(daesty[0]-daestz[0])*2/3;
                da[5]=daestz[5];
                da[6]=daesty[6];
                da[7]=daesty[7];
                da[8]=-(da[0]+da[4]);
                da[9]=(daesty[9]+daestz[9])/2;
                da[10]=daestz[10];
                da[11]=daesty[11];
        }

        else
        {

                if (mflag) {
                        //All 3 planes included:
                        //da[0]=(4*daesty[0]-daestx[0])+(daestx[0]+2*daestz[0])+(2*daesty[0]+daestz[0]);
                        da[0]=(2*daesty[0]+daestz[0])/3;
                        da[1]=(daesty[1]+daestz[1])/2;
                        da[2]=(daesty[2]+daestz[2])/2;
                        da[3]=(daestx[3]+daestz[3])/2;
                        //da[4]=(daestx[0]-daesty[0])*2+(daestx[0]-daestz[0])+(daesty[0]-daestz[0])*2;
                        da[4]=(daesty[0]-daestz[0])*2/3;
                        da[5]=(daestx[5]+daestz[5])/2;
                        da[6]=(daestx[6]+daesty[6])/2;
                        da[7]=(daestx[7]+daesty[7])/2;
                        da[8]=-(da[0]+da[4]);
                        da[9]=(daesty[9]+daestz[9])/2;
                        da[10]=(daestx[10]+daestz[10])/2;
                        da[11]=(daestx[11]+daesty[11])/2;
                }
        }

}

/*** Adjust average estimates for new source location and quadrupole strengths if electric field included. ***/
//There are three options:
//      1. Completely independent electric and magnetic quadrupoles but dipoles always averaged
//      2.      Average the diagonal elements only
//      3.      As for 2 but average the off diagonals in the electric matrix with the off diagonal
//              respective pairs of the magnetic matrix, noting that these will first have to be averaged
//      Note that dae and daeste are ordered as: Qxx Qxy Qxz Qyy Qyz for elements [0,1,2,3,4] and that Qxy, Qxz and Qyz are
// double their true values, and Qxx and Qyy are not their true values due to the linear dependence of Qzz


//Update with latest electric quadrupole and location estimate.
        if (ifp4 != NULL) {
                //Adjust electric quadrupoles back to their true values
                dae[0]=(2*daeste[0]-daeste[3])/3;
                dae[1]=daeste[1]/2;
                dae[2]=daeste[2]/2;
                dae[3]=(2*daeste[3]-daeste[0])/3;
                dae[4]=daeste[4]/2;
                dae[5]=daeste[5];
                dae[6]=daeste[6];
                dae[7]=daeste[7];

/*              for (i=0;i<8;i++) {
                                dae[i] = daeste[i];
                        }
*/
                //Set option required
                option=1;

                if (mflag) {
```

```
                    //always average the positions
                    for (i=0;i<6;i++) {
                              a[i] = (anewe[i] + a[i])/2;
                    }
                    //always average the dipole strengths
                    if (ind==1)
                    {
                              da[9] = dae[5];
                              da[10] = dae[6] = (da[10] + dae[6])/2;
                              da[11] = dae[7] = (da[11] + dae[7])/2;
                    }

                    else if (ind==3)
                    {
                              da[9] = dae[5]= (da[9] + dae[5])/2;;
                              da[10] = dae[6];
                              da[11] = dae[7] = (da[11] + dae[7])/2;
                    }


                    else if (ind==5)
                    {
                              da[9] = dae[5] = (da[9] + dae[5])/2;;
                              da[10] = dae[6] = (da[10] + dae[6])/2;
                              da[11] = dae[7];
                    }


                    else
                    {
                              da[9] = dae[5] = (da[9] + dae[5])/2;;
                              da[10] = dae[6] = (da[10] + dae[6])/2;
                              da[11] = dae[7] = (da[11] + dae[7])/2;;
                    }


                    if (option==2) {
                              da[0]=dae[0]=(da[0]+dae[0])/2;
                              da[4]=dae[3]=(da[4]+dae[3])/2;
                              da[8]=-(da[0]+da[4]);
                    }

                    else if (option==3) {
                              da[0]=dae[0]=(da[0]+dae[0])/2;
                              da[4]=dae[3]=(da[4]+dae[3])/2;
                              da[8]=-(da[0]+da[4]);
                              dae[1]=(dae[1]+(da[1]+da[3])/2)/2;
                              dae[2]=(dae[2]+(da[2]+da[6])/2)/2;
                              dae[4]=(dae[4]+(da[5]+da[7])/2)/2;
                    }

                    else ;

          }

          else {
                    //no magnetic planes
                    for (i=0;i<6;i++) {
                                                  a[i] = anewe[i];
                                        }
          }
}


/**** Volume currents' correction to magnetic field ****/
//          Using the new source strength estimates and best source location estimate, compute the volume correction
//          for the surface magnetic field.
//
//          There are a number of options for the quadrupole tensor which is used in the forward volume model
//
// If 2 or more magnetic planes and the electric field are included,
//          Option = 2 will use
//          a grand average of the quadrupole strengths.
//
//          Option = 1 will leave the electric and magnetic field
//          estimates separate.
//
```

```
// If a single magnetic plane and the electric field are included then use
//		Option = 1
//		This has no effect as the magnetic field case
//		produces incomplete information on the quadrupole tensor. The model uses the electric quadrupoles
//		for volume current estimate as it assumes a common source.
//
// If there is no electric field and a single magnetic plane then
//		Option = 3
//		will equate the relevant off-diagonals since in the volume model the off diagonal
//		functionals are linearly dependent. Note that if only a single magnetic plane is used (ie no elctric
//		field, this inverse method cannot find a convergent solution as the tensor diagonal is incomplete.


//Set option here
option=1;

Berrold=err[0];
err[0]=0;


if ((ind==1)||(ind==4)||(ind==6)||(ind==9))
{
/*		//If a single magnetic plane and electric potential then use electric quadrupoles only
		//for volume current model. This assumes that the source strengths are the same
		if ((ind==1) && (ifp4 != NULL))
		{

			dav[0]=dae[0]; dav[3]=dae[1]; dav[5]=dae[4]; dav[6]=dae[2]; dav[7]=dae[4];

			//Try this line instead if source strengths are different
			//dav[0]=da[0]; dav[3]=da[3]; dav[5]=da[5]; dav[6]=da[6]; dav[7]=da[7];

			dav[1]=dae[1];  dav[2]=dae[2]; dav[4]=dae[3];
			dav[8]=-(dav[0]+dav[4]);
		}

		else if (((ind==4)||(ind==6)||(ind==9))&&(ifp4 != NULL)&&(option==2))
		{
			//if 2 or more magnetic planes and electric potential then use a
			//grand average for volume current estimate
			dav[0]=(dae[0]+da[0])/2; dav[4]=(dae[3]+da[4])/2; dav[8]=-(dae[0]+dae[3]+da[0]+da[4])/2;
			dav[1]=dav[3]=(dae[1]+(da[1]+da[3])/2)/2; dav[2]=dav[6]=(dae[2]+(da[2]+da[6])/2)/2;
dav[5]=dav[7]=(dae[4]+(da[5]+da[7])/2)/2;

		}

		else
		{
			//Electric plane not included
			for (i=0;i<9;i++) dav[i]=da[i];
			//If only Bx plane included fill off diagonals dav[1] and dav[2]
			if ((ind==1)&&(option==3))
			{
				dav[1]=da[3]; dav[2]=da[6];
			}
		}


		fharmmcg2(0,a+3,Bx[2],BxA1,BxB1,dav,start,end,bc,mrx);

		//The new infinite medium estimate for the next iteration is the original
		//data less (ie added! see theory) the new volume correction for the quadrupole
*/
		for(i=0;i<mrx;i++)
		{
			//Bxnew[i]=Bx[1][i];
			//Bx[1][i]=Bx[0][i]+Bx[2][i];
			err[0] += pow((Bx[1][i]-Bxnew[i]),2);

		}

}

if ((ind==3)||(ind==4)||(ind==8)||(ind==9))
{
/*		//If single magnetic plane and electric potential then use electric quadrupoles only
		//for volume current estimate
		if ((ind==3) && (ifp4 != NULL))
```

```
                {
                        dav[0]=dae[0]; dav[1]=dae[1]; dav[2]=dae[2]; dav[6]=dae[2]; dav[7]=dae[4];

                        //Try this line instead if source strengths are different
                        //dav[0]=da[0]; dav[1]=da[1]; dav[2]=da[2]; dav[6]=da[6]; dav[7]=da[7];

                        dav[3]=dae[1]; dav[4]=dae[3]; dav[5]=dae[4];
                        dav[8]=-(dav[0]+dav[4]);
                }

                else if (((ind==4)||(ind==8)||(ind==9))&&(ifp4 != NULL)&&(option==2))
                {
                        //if 2 or more magnetic planes and electric potential then use a
                        //grand average for volume current estimate
                        dav[0]=(dae[0]+da[0])/2; dav[4]=(dae[3]+da[4])/2; dav[8]=-(dae[0]+dae[3]+da[0]+da[4])/2;
                        dav[1]=dav[3]=(dae[1]+(da[1]+da[3])/2)/2; dav[2]=dav[6]=(dae[2]+(da[2]+da[6])/2)/2;
dav[5]=dav[7]=(dae[4]+(da[5]+da[7])/2)/2;

                }

                else
                {
                        //Electric plane not included
                        for (i=0;i<9;i++) dav[i]=da[i];
                        //If only By plane included fill off diagonals dav[3] and dav[5]
                        if ((ind==4)&&(option==3))
                        {
                                dav[3]=da[1]; dav[5]=da[7];
                        }
                }

                fharmmcg2(1,a+3,By[2],ByA1,ByB1,dav,start,end,bc,mry);

                //The new infinite medium estimate for the next iteration is the original
                //data less (ie added! see theory) the new volume correction for the quadrupole
*/
                for(i=0;i<mry;i++)
                {
                        //Bynew[i]=By[1][i];
                        //By[1][i]=By[0][i]+By[2][i];
                        err[0] += pow((By[1][i]-Bynew[i]),2);
                }

        }

        if ((ind==5)||(ind==6)||(ind==8)||(ind==9))
        {
/*              //If single magnetic plane and electric potential then use electric quadrupoles only
                //for volume current estimate
                if ((ind==5) && (ifp4 != NULL))
                {
                        dav[0]=dae[0]; dav[1]=dae[1]; dav[2]=dae[2]; dav[3]=dae[1]; dav[5]=dae[4];

                        //Try this line instead if source strengths are different
                        //dav[0]=da[0]; dav[1]=da[1]; dav[2]=da[2]; dav[3]=da[3]; dav[5]=da[5];

                        dav[4]=dae[3]; dav[6]=dae[2]; dav[7]=dae[4];
                        dav[8]=-(dav[0]+dav[4]);
                }

                else if (((ind==6)||(ind==8)||(ind==9))&&(ifp4 != NULL)&&(option==2))
                {
                        //if 2 or more magnetic planes and electric potential then use a
                        //grand average for volume current estimate
                        dav[0]=(dae[0]+da[0])/2; dav[4]=(dae[3]+da[4])/2; dav[8]=-(dae[0]+dae[3]+da[0]+da[4])/2;
                        dav[1]=dav[3]=(dae[1]+(da[1]+da[3])/2)/2; dav[2]=dav[6]=(dae[2]+(da[2]+da[6])/2)/2;
dav[5]=dav[7]=(dae[4]+(da[5]+da[7])/2)/2;

                }

                else
                {
                        //Electric plane not included
                        for (i=0;i<9;i++) dav[i]=da[i];
                        //If only Bz plane included fill off diagonals dav[6] and dav[7]
                        if ((ind==6)&&(option==3))
                        {
```

```
                                        dav[6]=da[2]; dav[7]=da[5];
                                }
                        }


                        fharmmcg2(2,a+3,Bz[2],BzA1,BzB1,dav,start,end,bc,mrz);

                        //The new infinite medium estimate for the next iteration is the original
                        //data less (ie added! see theory) the new volume correction for the quadrupole
*/
                        for(i=0;i<mrz;i++)
                        {
                                //Bznew[i]=Bz[1][i];
                                //Bz[1][i]=Bz[0][i]+Bz[2][i];
                                err[0] += pow((Bz[1][i]-Bznew[i]),2);
                        }

                }


        free(ae);
        free(anewe);
        free(aeste);
        free(aestx);
        free(anew);
        free(aesty);
        free(aestz);
        free(anewacc);
        free(anewold);

        free(dav);
        free(daeste);
        free(daestx);
        free(daesty);
        free(daestz);

        free(phinew);
        free(Bxnew);
        free(Bynew);
        free(Bznew);


        return;
}
```

/** field1a**

Position dependent field constants for m=m'+1 and the m=m' series *****/

```
#include <stdio.h>
#include "nrutil.h"

void field1a (int flagxyz,float **BxB,float **BxA,float **Pe,float Cthe,float Sthe, float *Cmphe,float *Smphe,int start,int k)
{
        //Calculates position dependent field constants for the sphere
        //surface for one data point. Field1 computes the m=m'+1 series for
        //Ux and Uy and the m=m' series for Uz.
        //Note there is no m=0 term since the m' series can only start fom m'=0

        //k = number of summation points
        float       **Pdiff1,**fxA,**FXA,**fxB,**fyA,**fyB,**fzA,**FZA,**fzB,**fthA,**fthB,**fphA,**fphB,**fxthA,**FXTHA,**fxthB,
**fythA,**fythB,**fzthA,**fzthB,**fxphA,**fxphB,**fyphA,**fyphB,**fxphAalt,**FXPHAalt,
**fxphBalt,**fyphAalt,**fyphBalt,**fzphA,**fzphB,**fzphAalt,**fzphBalt,**furA,**furB,**futhA,**futhB,
**fuphA,**fuphB,Bxa=0,Bxb=0,Bya=0,Byb=0,Bza=0,Bzb=0;
        int         m,n;

        n=k+2;
        Pdiff1=make2darray(&n,&n);
        FXA=make2darray(&k,&k);
        FXTHA=make2darray(&k,&k);
        FXPHAalt=make2darray(&k,&k);
        FZA=make2darray(&k,&k);
        fxA=make2darray(&k,&k);
        fxB=make2darray(&k,&k);
        fyA=make2darray(&k,&k);
        fyB=make2darray(&k,&k);
        fzA=make2darray(&k,&k);
        fzB=make2darray(&k,&k);
        fthA=make2darray(&k,&k);
        fthB=make2darray(&k,&k);
        fphA=make2darray(&k,&k);
        fphB=make2darray(&k,&k);
        fxthA=make2darray(&k,&k);
        fxthB=make2darray(&k,&k);
        fythA=make2darray(&k,&k);
        fythB=make2darray(&k,&k);
        fzthA=make2darray(&k,&k);
        fzthB=make2darray(&k,&k);
        fxphA=make2darray(&k,&k);
        fxphB=make2darray(&k,&k);
        fyphA=make2darray(&k,&k);
        fyphB=make2darray(&k,&k);
        fxphAalt=make2darray(&k,&k);
        fxphBalt=make2darray(&k,&k);
        fyphAalt=make2darray(&k,&k);
        fyphBalt=make2darray(&k,&k);
        fzphA=make2darray(&k,&k);
        fzphB=make2darray(&k,&k);
        fzphAalt=make2darray(&k,&k);
        fzphBalt=make2darray(&k,&k);
        furA=make2darray(&k,&k);
        furB=make2darray(&k,&k);
        futhA=make2darray(&k,&k);
        futhB=make2darray(&k,&k);
        fuphA=make2darray(&k,&k);
        fuphB=make2darray(&k,&k);


        //Compute Legendre differentials
          for (n=0;n<=(k+2);n++){
            for (m=0;m<=n;m++){
                    if (m==0) Pdiff1[n][m] = -Pe[n][1];
                    else if (n<=(k+2)) {
                                    //*old method* Pdiff1[n][m]=-(n+1)*Pe[n][m]*Cthe/Sthe + (n-m+1)*Pe[n+1][m]/Sthe;
                                    Pdiff1[n][m]=((n+m)*(n-m+1)*Pe[n][m-1])/2;
                                    if (m<n) Pdiff1[n][m] -= (Pe[n][m+1])/2;
                            }
                    }
                  }
```

```
//Compute fx and fxth
for (n=1;n<=k;n++) {
/*** m starts from 1 in series 1 for x and y ******/
        for (m=1;m<=n;m++) {

                if (m==0) {
                        fxA[n][0] =  Pe[n+1][1]/(2*n+3) - Pe[n-1][1]/(2*n-1);
                }
                else fxA[n][m] = Pe[n-1][m-1]*(n+m)*(n+m-1)/(2*n-1) - Pe[n+1][m-1]*(n-m+2)*(n-m+1)/(2*n+3);

                /*if ((n>=3)&&((n-3)>=(m-1))) {
                fxA[n][m]+=Pe[n-3][m-1]*(n+m-3)*(n+m)*(n+m-1)*(n+m-2)*(n+m-3)/((n-m)*(n-m-1));
                }
                */

                fxA[n][m] /= (2*(2*n+1));
                FXA[n][m] = fxA[n][m];
                fxB[n][m] = fxA[n][m];
                fyA[n][m] = fxA[n][m];
                fyB[n][m] = fxA[n][m];

                fxphA[n][m] = fxA[n][m];
                fxphB[n][m] = fxA[n][m];
                fyphA[n][m] = fxA[n][m];
                fyphB[n][m] = fxA[n][m];

                if (m==0) {
                fxA[n][m] *= Cmphe[1];
                fyA[n][m] *= Cmphe[1];
                fxB[n][m] *= -Smphe[1];
                fyB[n][m] *= Smphe[1];

                fxphA[n][m] *= (m-1)*Smphe[1];
                fyphA[n][m] *= (m-1)*Smphe[1];
                fxphB[n][m] *= (m-1)*Cmphe[1];
                fyphB[n][m] *= -(m-1)*Cmphe[1];
                }
                else {
                        fxA[n][m] *= Cmphe[m-1];
                        fyA[n][m] *= Cmphe[m-1];
                        fxB[n][m] *= Smphe[m-1];
                        fyB[n][m] *= -Smphe[m-1];

                        fxphA[n][m] *= -(m-1)*Smphe[m-1];
                        fyphA[n][m] *= -(m-1)*Smphe[m-1];
                        fxphB[n][m] *= (m-1)*Cmphe[m-1];
                        fyphB[n][m] *= -(m-1)*Cmphe[m-1];
                }

                /****** f_ph_alt alternative version*/
                //Note that the following expressions for f_ph all contain a factor
                //Sthe/(Cthe*(m-1)) but which is excluded as it later cancels
                if (m==0) {
                        fxphAalt[n][0] = (Pe[n-1][0]*(n)*(n-1) + Pe[n-1][2])/(2*(2*n-1)) - (Pe[n+1][0]*(n+2)*(n+1) +
Pe[n+1][2])/(2*(2*n+3));
                }

                else if (m==1) {
                        fxphAalt[n][1] = 0;
                }
                else {
                        fxphAalt[n][m] = (Pe[n-1][m] + Pe[n-1][m-2]*(n+m-2)*(n-m+1))*(n+m)*(n+m-1)/(2*(2*n-1)) -
                                (Pe[n+1][m] + Pe[n+1][m-2]*(n+m)*(n-m+3))*(n-m+2)*(n-m+1)/(2*(2*n+3));
                }


                fxphAalt[n][m] /= (2*(2*n+1));
                FXPHAalt[n][m] = fxphAalt[n][m];
                fxphBalt[n][m] = fxphAalt[n][m];
                fyphAalt[n][m] = fxphAalt[n][m];
                fyphBalt[n][m] = fxphAalt[n][m];

                //(m-1) is excluded here

                if (m==0) {
                        fxphAalt[n][m] *= Smphe[1];
```

```
                        fyphAalt[n][m] *= Smphe[1];
                        fxphBalt[n][m] *= Cmphe[1];
                        fyphBalt[n][m] *= -Cmphe[1];
                }

                else {
                        fxphAalt[n][m] *= -Smphe[m-1];
                        fyphAalt[n][m] *= -Smphe[m-1];
                        fxphBalt[n][m] *= Cmphe[m-1];
                        fyphBalt[n][m] *= -Cmphe[m-1];
                }


                if (m==0) fxthA[n][0] = (Pe[n-1][2] - Pe[n-1][0]*(n)*(n-1))/(2*(2*n-1)) - (Pe[n+1][2] -
Pe[n+1][0]*(n+2)*(n+1))/(2*(2*n+3));
                else fxthA[n][m] = Pdiff1[n-1][m-1]*(n+m)*(n+m-1)/(2*n-1) - Pdiff1[n+1][m-1]*(n-m+2)*(n-m+1)/(2*n+3);

                fxthA[n][m] /= (2*(2*n+1));
                FXTHA[n][m] = fxthA[n][m];
                fxthB[n][m] = fxthA[n][m];
                fythA[n][m] = fxthA[n][m];
                fythB[n][m] = fxthA[n][m];

                if (m==0) {
                        fxthA[n][0] *= Cmphe[1];
                        fythA[n][0] *= Cmphe[1];
                        fxthB[n][0] *= -Smphe[1];
                        fythB[n][0] *= Smphe[1];
                }
                else {
                        fxthA[n][m] *= Cmphe[m-1];
                        fythA[n][m] *= Cmphe[m-1];
                        fxthB[n][m] = Smphe[m-1];
                        fythB[n][m] *= -Smphe[m-1];
                }

        }
}

//Compute fz and fzth
for (n=1;n<=k;n++) {
        for (m=0;m<=n;m++) {
                fzA[n][m] = Pe[n+1][m]*(n-m+1)/(2*n+3);
                if ((n-1)>=0) fzA[n][m] += Pe[n-1][m]*(n+m)/(2*n-1);


                FZA[n][m] = fzA[n][m];
                fzB[n][m] = fzA[n][m];
                fzphA[n][m] = fzA[n][m];
                fzphB[n][m] = fzA[n][m];

                fzA[n][m] *= Cmphe[m]/(2*n+1);
                fzB[n][m] *= Smphe[m]/(2*n+1);
                fzphA[n][m] *= -m*Smphe[m]/(2*n+1);
                fzphB[n][m] *= m*Cmphe[m]/(2*n+1);

                /****** f_ph_alt alternative version*/
                //Note that the following expressions for f_ph all contain a factor
                //Sthe/(Cthe*m) but which is excluded as it later cancels
                if (m==0) {
                        fzphAalt[n][m]=0;
                }
                else {
                        if (n!=0)fzphAalt[n][m] = (Pe[n-1][m+1] + Pe[n-1][m-1]*(n+m-1)*(n-m))*(n+m)/(2*(2*n-1));
                        fzphAalt[n][m] += (Pe[n+1][m+1] + Pe[n+1][m-1]*(n+m+1)*(n-m+2))*(n-m+1)/(2*(2*n+3));
                }

                fzphBalt[n][m]=fzphAalt[n][m];
                //m is excluded here
                fzphAalt[n][m] *= -Smphe[m]/(2*n+1);
                fzphBalt[n][m] *= Cmphe[m]/(2*n+1);

                fzthA[n][m] = Pdiff1[n+1][m]*(n-m+1)/(2*n+3);
                if (n!=0) fzthA[n][m] += Pdiff1[n-1][m]*(n+m)/(2*n-1);

                fzthB[n][m] = fzthA[n][m];

                fzthA[n][m] *= Cmphe[m]/(2*n+1);
```

```
                fzthB[n][m] *= Smphe[m]/(2*n+1);

        }
}



//fur, futh and fuph are intermediates
for (n=start;n<=k;n++) {
        for (m=0;m<=n;m++) {
                fthA[n][m] = fxA[n][m]*Cthe*Cmphe[1] + fyA[n][m]*Cthe*Smphe[1] - fzA[n][m]*Sthe;
                fphA[n][m] = -fxA[n][m]*Smphe[1] + fyA[n][m]*Cmphe[1];

                fthB[n][m] = fxB[n][m]*Cthe*Cmphe[1] + fyB[n][m]*Cthe*Smphe[1] - fzB[n][m]*Sthe;
                fphB[n][m] = -fxB[n][m]*Smphe[1] + fyB[n][m]*Cmphe[1];

                //Cthe/Sthe is excluded here in the f_ph_alt expression
                furA[n][m] = -fxthA[n][m]*Smphe[1] + fythA[n][m]*Cmphe[1] -
                                        (fxphA[n][m]*Cmphe[1] + fyphA[n][m]*Smphe[1])*Cthe/Sthe + fzphA[n][m];

                //Cthe/Sthe is excluded here in the fzphAalt expression
                futhA[n][m] = (n)*fphA[n][m] + fxphA[n][m]*Cmphe[1] + fyphA[n][m]*Smphe[1] + fzphA[n][m]*Cthe/Sthe;

                fuphA[n][m] = -(n)*fthA[n][m] -fxthA[n][m]*Cmphe[1]*Sthe - fythA[n][m]*Smphe[1]*Sthe - fzthA[n][m]*Cthe;

                //Cthe/Sthe is excluded here in the f_ph_alt expression
                furB[n][m] =  -fxthB[n][m]*Smphe[1] + fythB[n][m]*Cmphe[1] -
                                        (fxphB[n][m]*Cmphe[1] + fyphB[n][m]*Smphe[1])*Cthe/Sthe + fzphB[n][m];

                //Cthe/Sthe is excluded here in the fzphBalt expression
                futhB[n][m] =           (n)*fphB[n][m] + fxphB[n][m]*Cmphe[1] + fyphB[n][m]*Smphe[1] + fzphB[n][m]*Cthe/Sthe;

                fuphB[n][m] = -(n)*fthB[n][m] -fxthB[n][m]*Cmphe[1]*Sthe - fythB[n][m]*Smphe[1]*Sthe - fzthB[n][m]*Cthe;

                //and finally Bx, By and Bz odd and even

                if (flagxyz==0)
                {
                        BxA[n][m] = (furA[n][m]*Sthe + futhA[n][m]*Cthe)*Cmphe[1] - fuphA[n][m]*Smphe[1];
                        BxB[n][m] = (furB[n][m]*Sthe + futhB[n][m]*Cthe)*Cmphe[1] - fuphB[n][m]*Smphe[1] ;

                }
                else if (flagxyz==1) //the ByA and B terms
                {
                        BxA[n][m] = (furA[n][m]*Sthe + futhA[n][m]*Cthe)*Smphe[1] + fuphA[n][m]*Cmphe[1];
                        BxB[n][m] = (furB[n][m]*Sthe + futhB[n][m]*Cthe)*Smphe[1] + fuphB[n][m]*Cmphe[1];
                }
                else //the BzA and B terms
                {
                        BxA[n][m] = furA[n][m]*Cthe - futhA[n][m]*Sthe;
                        BxB[n][m] = furB[n][m]*Cthe - futhB[n][m]*Sthe;
                }


        }
}

n=k+2;
free2darray(Pdiff1,&n);
free2darray(FXA,&k);
free2darray(FXTHA,&k);
free2darray(FXPHAalt,&k);
free2darray(FZA,&k);
free2darray(fxA,&k);
free2darray(fxB,&k);
free2darray(fyA,&k);
free2darray(fyB,&k);
free2darray(fzA,&k);
free2darray(fzB,&k);
free2darray(fthA,&k);
free2darray(fthB,&k);
free2darray(fphA,&k);
free2darray(fphB,&k);
free2darray(fxthA,&k);
free2darray(fxthB,&k);
free2darray(fythA,&k);
```

```
    free2darray(fythB,&k);
    free2darray(fzthA,&k);
    free2darray(fzthB,&k);
    free2darray(fxphA,&k);
    free2darray(fxphB,&k);
    free2darray(fyphA,&k);
    free2darray(fyphB,&k);
    free2darray(fzphA,&k);
    free2darray(fzphB,&k);
    free2darray(fxphAalt,&k);
    free2darray(fxphBalt,&k);
    free2darray(fyphAalt,&k);
    free2darray(fyphBalt,&k);
    free2darray(fzphAalt,&k);
    free2darray(fzphBalt,&k);
    free2darray(furA,&k);
    free2darray(furB,&k);
    free2darray(futhA,&k);
    free2darray(futhB,&k);
    free2darray(fuphA,&k);
    free2darray(fuphB,&k);

    return;
}
```

/** field2a**

Position dependent field constants for m=m'-1 series ****/

```
#include <stdio.h>
#include "nrutil.h"

void field2a (int flagxyz,float **BxB,float **BxA,float **Pe,float Cthe,float Sthe, float *Cmphe,float *Smphe,int start,int k)
{
        //Calculates position dependent field constants for the sphere
        //surface for one data point. Field2 computes the m=m'-1 series
        //for Ux and Uy

        //k = number of summation points
        float       **Pdiff1,**fxA,**FXA,**fxB,**fyA,**fyB,**fthA,**fthB,**fphA,**fphB,**fxthA,**FXTHA,**fxthB,
**fythA,**fythB,**fxphA,**fxphB,**fyphA,**fyphB,**fxphAalt,**FXPHAalt,**fxphBalt,**fyphAalt,
**fyphBalt,**furA,**furB,**futhA,**futhB,**fuphA,**fuphB,Bxa=0,Bxb=0,Bya=0,Byb=0,Bza=0,Bzb=0;
        int         m,n;

        n=k+2;
        Pdiff1=make2darray(&n,&n);
        FXA=make2darray(&k,&k);
        FXTHA=make2darray(&k,&k);
        FXPHAalt=make2darray(&k,&k);
        fxA=make2darray(&k,&k);
        fxB=make2darray(&k,&k);
        fyA=make2darray(&k,&k);
        fyB=make2darray(&k,&k);
        fthA=make2darray(&k,&k);
        fthB=make2darray(&k,&k);
        fphA=make2darray(&k,&k);
        fphB=make2darray(&k,&k);
        fxthA=make2darray(&k,&k);
        fxthB=make2darray(&k,&k);
        fythA=make2darray(&k,&k);
        fythB=make2darray(&k,&k);
        fxphA=make2darray(&k,&k);
        fxphB=make2darray(&k,&k);
        fyphA=make2darray(&k,&k);
        fyphB=make2darray(&k,&k);
        fxphAalt=make2darray(&k,&k);
        fxphBalt=make2darray(&k,&k);
        fyphAalt=make2darray(&k,&k);
        fyphBalt=make2darray(&k,&k);
        furA=make2darray(&k,&k);
        furB=make2darray(&k,&k);
        futhA=make2darray(&k,&k);
        futhB=make2darray(&k,&k);
        fuphA=make2darray(&k,&k);
        fuphB=make2darray(&k,&k);


        //Compute Legendre differentials
         for (n=0;n<=(k+2);n++){
           for (m=0;m<=n;m++){
                    if (m==0) Pdiff1[n][m] = -Pe[n][1];
                    else if (n<=(k+2)) {
                                        //*old method* Pdiff1[n][m]=-(n+1)*Pe[n][m]*Cthe/Sthe + (n-m+1)*Pe[n+1][m]/Sthe;
                                        Pdiff1[n][m]=((n+m)*(n-m+1)*Pe[n][m-1])/2;
                                        if (m<n) Pdiff1[n][m] -= (Pe[n][m+1])/2;
                              }
                   }
                  }


        //Compute fx and fxph
        for (n=1;n<=k;n++) {
                    for (m=0;m<=n;m++) {

                              fxA[n][m] = Pe[n+1][m+1]/(2*n+3);
                              if ((m+1)<=(n-1)) fxA[n][m] -= Pe[n-1][m+1]/(2*n-1);

                              //fxA[n][m]+=Pe[n+2][m+1]*(n-m+1)*(n-m)/((n+m+3)*(n+m+2)*(n+m+1));

                              fxA[n][m] /= (2*(2*n+1));
                              FXA[n][m] = fxA[n][m];
```

```
                fxB[n][m] = fxA[n][m];
                fyA[n][m] = fxA[n][m];
                fyB[n][m] = fxA[n][m];

                fxphA[n][m] = fxA[n][m];
                fxphB[n][m] = fxA[n][m];
                fyphA[n][m] = fxA[n][m];
                fyphB[n][m] = fxA[n][m];

                if (m==0) {
                fxA[n][m] *= 2*Cmphe[1];
                fyA[n][m] *= 0;
                fxB[n][m] *= 0;
                fyB[n][m] *= 2*Smphe[1];

                fxphA[n][m] *= -2*Smphe[1];
                fyphA[n][m] *= 0;
                fxphB[n][m] *= 0;
                fyphB[n][m] *= 2*Cmphe[1];
                }
                else {
                        fxA[n][m] *= Cmphe[m+1];
                        fyA[n][m] *= -Cmphe[m+1];
                        fxB[n][m] *= Smphe[m+1];
                        fyB[n][m] *= Smphe[m+1];

                        fxphA[n][m] *= -(m+1)*Smphe[m+1];
                        fyphA[n][m] *= (m+1)*Smphe[m+1];
                        fxphB[n][m] *= (m+1)*Cmphe[m+1];
                        fyphB[n][m] *= (m+1)*Cmphe[m+1];
                }

                /****** f_ph_alt alternative version*/
                //Note that the following expressions for f_ph all contain a factor
                //Sthe/(Cthe*(m+1)) but which is excluded as it later cancels

                fxphAalt[n][m] = (Pe[n+1][m+2] + Pe[n+1][m]*(n+m+2)*(n-m+1))/(2*(2*n+3));
                if((n-1)>=0) fxphAalt[n][m] -= (Pe[n-1][m+2] + Pe[n-1][m]*(n+m)*(n-m-1))/(2*(2*n-1));


                fxphAalt[n][m] /= (2*(2*n+1));
                FXPHAalt[n][m] = fxphAalt[n][m];
                fxphBalt[n][m] = fxphAalt[n][m];
                fyphAalt[n][m] = fxphAalt[n][m];
                fyphBalt[n][m] = fxphAalt[n][m];

                //(m+1) is excluded here

                        fxphAalt[n][m] *= -Smphe[m+1];
                        fyphAalt[n][m] *= Smphe[m+1];
                        fxphBalt[n][m] *= Cmphe[m+1];
                        fyphBalt[n][m] *= Cmphe[m+1];

                fxthA[n][m] = Pdiff1[n+1][m+1]/(2*n+3);
                if ((m+1)<=(n-1)) fxthA[n][m] -= Pdiff1[n-1][m+1]/(2*n-1);


                //fxthA[n][m]+=Pdiff1[n+2][m+1]*(n-m+1)*(n-m)/((n+m+3)*(n+m+2)*(n+m+1));

                fxthA[n][m] /= (2*(2*n+1));
                FXTHA[n][m] = fxthA[n][m];
                fxthB[n][m] = fxthA[n][m];
                fythA[n][m] = fxthA[n][m];
                fythB[n][m] = fxthA[n][m];

                fxthA[n][m] *= Cmphe[m+1];
                fythA[n][m] *= -Cmphe[m+1];
                fxthB[n][m] *= Smphe[m+1];
                fythB[n][m] *= Smphe[m+1];

        }
}

//fur, futh and fuph are intermediates
for (n=start;n<=k;n++) {
        for (m=0;m<=n;m++) {
                fthA[n][m] = fxA[n][m]*Cthe*Cmphe[1] + fyA[n][m]*Cthe*Smphe[1];
                fphA[n][m] = -fxA[n][m]*Smphe[1] + fyA[n][m]*Cmphe[1];
```

```
                fthB[n][m] = fxB[n][m]*Cthe*Cmphe[1] + fyB[n][m]*Cthe*Smphe[1];
                fphB[n][m] = -fxB[n][m]*Smphe[1] + fyB[n][m]*Cmphe[1];

                //Cthe/Sthe is excluded here in the f_ph_alt expression
                furA[n][m] = -fxthA[n][m]*Smphe[1] + fythA[n][m]*Cmphe[1] -
                                        (fxphA[n][m]*Cmphe[1] + fyphA[n][m]*Smphe[1])*Cthe/Sthe ;

                //Cthe/Sthe is excluded here in the fzphAalt expression
                futhA[n][m] = (n)*fphA[n][m] + fxphA[n][m]*Cmphe[1] + fyphA[n][m]*Smphe[1];

                fuphA[n][m] = -(n)*fthA[n][m] -fxthA[n][m]*Cmphe[1]*Sthe - fythA[n][m]*Smphe[1]*Sthe ;

                //Cthe/Sthe is excluded here in the f_ph_alt expression
                furB[n][m] = -fxthB[n][m]*Smphe[1] + fythB[n][m]*Cmphe[1] -
                                        (fxphB[n][m]*Cmphe[1] + fyphB[n][m]*Smphe[1])*Cthe/Sthe;

                //Cthe/Sthe is excluded here in the fzphBalt expression
                futhB[n][m] =  (n)*fphB[n][m] + fxphB[n][m]*Cmphe[1] + fyphB[n][m]*Smphe[1];

                fuphB[n][m] = -(n)*fthB[n][m] -fxthB[n][m]*Cmphe[1]*Sthe - fythB[n][m]*Smphe[1]*Sthe;

                //and finally Bx, By or Bz odd and even
                if (flagxyz==0)
                {
                        BxA[n][m] = (furA[n][m]*Sthe + futhA[n][m]*Cthe)*Cmphe[1] - fuphA[n][m]*Smphe[1];
                        BxB[n][m] = (furB[n][m]*Sthe + futhB[n][m]*Cthe)*Cmphe[1] - fuphB[n][m]*Smphe[1] ;

                }
                else if (flagxyz==1)  //the ByA and B terms
                {
                        BxA[n][m] = (furA[n][m]*Sthe + futhA[n][m]*Cthe)*Smphe[1] + fuphA[n][m]*Cmphe[1];
                        BxB[n][m] = (furB[n][m]*Sthe + futhB[n][m]*Cthe)*Smphe[1] + fuphB[n][m]*Cmphe[1];

                }
                else //the BzA and B terms
                {
                        BxA[n][m] = furA[n][m]*Cthe - futhA[n][m]*Sthe;
                        BxB[n][m] = furB[n][m]*Cthe - futhB[n][m]*Sthe;
                }

        }
}

n=k+2;
free2darray(Pdiff1,&n);
free2darray(FXA,&k);
free2darray(FXTHA,&k);
free2darray(FXPHAalt,&k);
free2darray(fxA,&k);
free2darray(fxB,&k);
free2darray(fyA,&k);
free2darray(fyB,&k);
free2darray(fthA,&k);
free2darray(fthB,&k);
free2darray(fphA,&k);
free2darray(fphB,&k);
free2darray(fxthA,&k);
free2darray(fxthB,&k);
free2darray(fythA,&k);
free2darray(fythB,&k);
free2darray(fxphA,&k);
free2darray(fxphB,&k);
free2darray(fyphA,&k);
free2darray(fyphB,&k);
free2darray(fxphAalt,&k);
free2darray(fxphBalt,&k);
free2darray(fyphAalt,&k);
free2darray(fyphBalt,&k);
free2darray(furA,&k);
free2darray(furB,&k);
free2darray(futhA,&k);
free2darray(futhB,&k);
free2darray(fuphA,&k);
free2darray(fuphB,&k);

return;
}
```

```
/**** HHdq2project

        The Electric Potential Inverse ****/


#include <stdio.h>
#include <stdlib.h>
#include <console.h>
#include <ctype.h>
#include <string.h>
#include <math.h>
#include "nrutil.h"
#include "in_data.h"

void orthog (float **,float **,int *,int *,int *,float *);
void fquad4ecg(int,float *,float *,float *,float *,int,float *);
void gaussj2(float **, int , float **, int );
void housch(float **,int *,int *,int *,int *,float **,float *,int *,float *);


void HHdq2project       (float **A,int mr,int nc,int rank,float*a,float *aest,float *da,
                                        float *daest,float *phi,float *phinew,float vj)
{

        /*** Electric Variable Projection Inverse ***/
        //This version is for combined dipole and quadrupole but separate locations

        float       *Qphi,**b,**v,**Q,**T11Q,w,*wd,**vtv,**dvda,**T11,*rbeta,precision=0.001;
        int         mrmat,ncmat,rankmat,*perm,sbreak,startrow,kay;
        register int  i,j,k;

        /*      Variables are:
                        mr long  vectors of m data points
                                phi     -       measured/true surface potential
                                rbeta   -       Householder intermediate value
                                Qphi    -       Q x phi = y bar

                        6 element vectors   [0],[1] and [2] are dipole [3],[4] and [5] are quadrupole
                                a       -       source coordinates used to generate the data
                                aest    -       estimated increment of source coordinates


                        nc element vector [0] to [4] = quadrupole   [5] to [7] = dipole
                                da      -       quadrupole and dipole vector strengths used to generate the data
                                daest   -       estimated quadrupole vector strengths
                                perm    -       permutation vector set up in the householder decomposition.
                                                sbreak indicates pseudo rank deficiency point as controlled
                                                by precision.

                        rank x rank matrix
                                T11     -       orthog transformed upper triang. or its inverse.

                        rank x (mr-rank) matrix
                                T11Q    -       orthogonal complement projector for the source
                                                        position increments

                        mr x rank matrices (in the combined case 'rank'= 8 = nc non-linear functionals
                                                but rank changes to 6 when the x,y,z coordinates are estimated fro each pole)
                                A       -       mr surface coordinates

                        mr x nc matrices (where nc=8 non-linear functionals)
                                v       -       mr sets non-linear functionals. The surface potential due to
                                                each multipole.  Note that these are arranged as Fxx, Fxy, Fxz, Fyy, Fyz, Fx, Fy, Fz
                                dvda    -       mr sets of partial differentials of v wrt to each a.
                                vtv     -       basis columns for v from Householder

                        mr x mr matrix
                                Q       -       orthogonal projector – not used in later versions
        */
        mrmat=mr-1;
        ncmat=0;
        rankmat=rank-1;
        //b is set up as one column array to allow gaussj2 to work
        b=make2darray(&rankmat,&ncmat);


        ncmat=nc-1;
```

```
dvda=make2darray(&mrmat,&ncmat);
//Q=make2darray(&mrmat,&mrmat);
Qphi=malloc((mr)*sizeof(float));
T11=make2darray(&rankmat,&rankmat);

v=make2darray(&mrmat,&ncmat);
vtv=make2darray(&mrmat,&ncmat);

ncmat=mr-rank-1;
T11Q=make2darray(&rankmat,&ncmat);

perm=malloc((nc)*sizeof(int));
wd=malloc((nc)*sizeof(float));
rbeta=malloc((nc)*sizeof(float));

for (i=0;i<mr;i++) {

        //Dipole functionals are added on at [5] [6] and [7]
        fquad4ecg(1,*(A+i),a,*(v+i),*(dvda+i),nc,da);

        phinew[i] = 0;
        for (j=0;j<rank;j++){
                phinew[i] +=da[j]*v[i][j];
                }
        }

househ (v,&mr,&nc,&rank,perm,vtv,&precision,&sbreak,rbeta);

for (i=0;i<mr;i++)
        {
                Qphi[i]=phi[i];
        }

startrow=0;
for (j=0;j<=sbreak;j++) {
        w=0.0;
        for (i=startrow;i<mr;i++) {
                w += vtv[i][perm[j]]*Qphi[i];
        }
        w=w*rbeta[perm[j]];
        for (i=startrow;i<mr;i++) {
                Qphi[i] +=w*vtv[i][perm[j]];
        }
        if (startrow<(rank-1)) startrow++;
}


/*      for (i=0;i<mr;i++)
        {
                for (j=0;j<nc;j++) printf("%8.4f ",v[i][j]);
                printf("\n");
        }
*/

        //Create T11 and then inverse

        for (i=0;i<rank;i++) {
                for (j=0;j<rank;j++) T11[i][j]=v[i][perm[j]];
        }

/*      printf("\n");
        for (i=0;i<rank;i++)
        {
                for (j=0;j<rank;j++) printf("%8.4f ",T11[i][j]);
                printf("\n");
        }
*/
        gaussj2(T11,rank,b,1);

/*      for (i=0;i<rank;i++)
        {
                for (j=0;j<rank;j++) printf("%8.4f ",T11[i][j]);
                printf("\n");
        }
*/
        //Now compute and display dipole and quadrupole strengths re-permutated back
        for (i=0;i<rank;i++) {
                dacst[perm[i]]=0.0;
```

```
                for (j=0;j<rank;j++) daest[perm[i]] +=T11[i][j]*Qphi[j];
        }

        //printf("Dx=%f Dy=%f Dz=%f Qxx=%f Qxy=%f Qxz=%f Qyy=%f Qzy=%f\n",daest[5],daest[6],daest[7],
//daest[0],daest[1],daest[2],daest[3],daest[4]);
/*      for (j=0;j<rank;j++){
                daest[j] =da[j];
        }
*/

        /* Now find the x, y and z increments */

        //The estimated Quadrupole strengths x the partial derivs wrt each non-lin var.
        //These are summed wrt d/dx  d/dy and d/dz respectively to give an mr x 3 matrix

        kay=6; //for the tensor arithmetic

        for (i=0;i<mr;i++) {
                //Diffs wrt to dipole location are in [0] [1] and [2], and
                //diffs wrt to quadrupole location are in [3] [4] and [5].
                fquad4ecg(0,*(A+i),a,*(v+i),*(dvda+i),nc,daest);
                for (j=0;j<kay;j++) {
                        v[i][j]=-dvda[i][j]; //Require the negative orthog dvda
                }
        }
/*
        for (i=0;i<mr;i++)
        {
                for (j=0;j<kay;j++) printf("%8.4f ",v[i][j]);
                printf("\n");
        }
        printf("\n");
*/
        //These are multiplied by the negative orthog matrix but starting
        // from (rankmat + 1) to form a new matrix v

        startrow=0;
        for (j=0;j<=sbreak;j++) {
                for(k=0;k<kay;k++) {
                        wd[k]=0.0;
                        for (i=startrow;i<mr;i++) {
                                wd[k] += vtv[i][perm[j]]*v[i][k];
                        }
                        wd[k]=wd[k]*rbeta[perm[j]];
                        for (i=startrow;i<mr;i++) {
                                v[i][k] +=wd[k]*vtv[i][perm[j]];
                                //printf("%8.4f \n",v[i][k]);
                        }
                        //printf("\n");
                }
                if (startrow<(rank-1)) startrow++;
        }

/*      for (i=0;i<mr-5;i++)
        {
                for (j=0;j<kay;j++) printf("%8.4f ",v[i][j]);
                printf("\n");
        }
        printf("\n");
*/
        //Take the mr-rank rows
        for (i=rank;i<mr;i++) {
                for (j=0;j<kay;j++) {
                        v[i-rank][j]=v[i][j];
                }
        }
        //Set the remainder of v to zero
        for (i=mr-rank;i<mr;i++)
                for (j=0;j<kay;j++) v[i][j]=0;

/*      for (i=0;i<mr-5;i++)
        {
                for (j=0;j<kay;j++) printf("%8.4f ",v[i][j]);
                printf("\n");
        }
        printf("\n");
*/
        //Now add upper triang Cholesky factor to the end of v. This version
```

```
        // asks for a control factor which is used as a multiplier on an identity matrix
        for (i=mr-kay;i<mr;i++) {
                for (j=0;j<kay;j++) {
                        if ((i-(mr-kay)) == j) v[i][j] = vj;
                        else v[i][j] = 0;
                }
        }

/*      for (i=0;i<mr-5;i++)
        {
                for (j=0;j<kay;j++) printf("%8.4f ",v[i][j]);
                printf("\n");
        }
        printf("\n");
*/
        nc=6;
        househ (v,&mr,&nc,&kay,perm,vtv,&precision,&sbreak,rbeta);

/*      printf("\n");
        for (i=0;i<mr-5;i++)
        {
                for (j=0;j<nc;j++) printf("%8.4f ",v[i][j]);
                printf("\n");
        }
*/

        //Remove the first rank elements of Qphi and add rank zeros
        for (i=0;i<mr;i++) {
                if (i<(mr-rank)) Qphi[i]=Qphi[i+rank];
                else Qphi[i]=0;
        }
        //Now orthogonalize
        startrow=0;
        for (j=0;j<=sbreak;j++) {
                w=0.0;
                for (i=startrow;i<mr;i++) {
                        w += vtv[i][perm[j]]*Qphi[i];
                }
                w=w*rbeta[perm[j]];
                for (i=startrow;i<mr;i++) {
                        Qphi[i] +=w*vtv[i][perm[j]];
                }
                if (startrow<(kay-1)) startrow++;
        }

/*      printf("\nQphi");
        for (i=0;i<mr-5;i++) printf("%f\n",Qphi[i]);
*/

        //Create T11 and then inverse

        for (i=0;i<kay;i++) {
                for (j=0;j<kay;j++) T11[i][j]=v[i][perm[j]];
        }

/*      printf("\n");
        for (i=0;i<kay;i++)
        {
                for (j=0;j<kay;j++) printf("%8.4f ",T11[i][j]);
                printf("\n");
        }
*/
        gaussj2(T11,kay,b,1);

/*      for (i=0;i<kay;i++)
        {
                for (j=0;j<kay;j++) printf("%8.4f ",T11[i][j]);
                printf("\n");
        }
*/
        //Multiply new Q by T11 inverse. Only the first kay rows of Q are
        //needed since T11 is a kay x kay matrix. T11Q is also multiplied
        // by the permutation matrix.
/**
        for (i=0;i<kay;i++){
                for (j=0;j<(mr-kay);j++) {
                        T11Q[perm[i]][j]=0;
                        for (k=0;k<kay;k++) T11Q[perm[i]][j] += T11[i][k]*Q[k][j];
```

```
                }
        }

        for (i=0;i<kay;i++)
        {
                for (j=0;j<(mr-kay);j++) printf("%8.4f ",T11Q[i][j]);
                printf("\n");
        }



        for (i=0;i<kay;i++){
                aest[i]=0;
                for (j=0;j<(mr-kay);j++) {
                        aest[i] += T11Q[i][j]*Qphi[j+kay];
                }
        }

**/
        for (i=0;i<kay;i++){
                aest[perm[i]]=0;
                for (j=0;j<(kay);j++) {
                        aest[perm[i]] += T11[i][j]*Qphi[j];
                }
        }



        //printf("\n\ndx=%8.5f dy=%8.5f dz=%8.5f",aest[0],aest[1],aest[2]);
        //printf("\n\ndx=%8.5f dy=%8.5f dz=%8.5f",aest[3],aest[4],aest[5]);
        //printf("\n");
        free2darray(v,&mrmat);
        free2darray(b,&rankmat);
        free2darray(dvda,&mrmat);
        free(Qphi);
        free2darray(T11,&rankmat);
        //free2darray(Q,&mrmat);
        free2darray(vtv,&mrmat);
        free2darray(T11Q,&rankmat);
        free(perm);
        free(wd);
        free(rbeta);

        return;
}
```

/** HHdqmag3proj**

Magnetic Field Inverse ****/

```
#include <stdio.h>
#include <stdlib.h>
#include <console.h>
#include <ctype.h>
#include <string.h>
#include <math.h>
#include "nrutil.h"
#include "in_data.h"


void fquad1mcg(float *,float *,float *);
void fharmmcg3b(int,float *,float **,float ***,float ***,float *,int,int,float **,int);
void fharmmcg3d(int,float *,float **,float ***,float ***,float *,int,int,float **,int);
void fdip0mcg(int,float *,float *,float *,float *);
void fdipmcg(float *,float *,float *);
void houseb(float **,int *,int *,int *,int *,float **,float *,int *,float *);
void gaussj2(float **, int , float **, int );


void HHdqmag3proj      (int flagxyz,int ind,float **A,int mr,float*a,float *aest,float *da,
float *daest,float *B,float *Bnew,float vj,int *qxyz,int *qxyzv,float ***BcA,float ***BcB,int start,int end,float **bc)
{

        float       *Qphi,**b,**v,**fv,**Q,**T11Q,w,*wd,**vtv,**dvda,**T11,*rbeta,precision=0.001;
        int          mrmat,ncmat,rankmat,*perm,sbreak,startrow,kay,xyz,rank=12,nc=12,idx=2;
        register int i,j,k;
        char fname5[]="::shndata:HHres";
/*      FILE *ifp9;

        if((ifp9=fopen(fname5,"a")) == NULL) {
                                printf("Cannot open file %s\n",fname5);
                                return;
        }
*/

        /*      Magnetic Variable Projection Inverse

                This version has separate dipole and quadrupole locations.
                Rank can be altered inside this function or in calling function.
                If the latter is used, exclusion vectors may be fixed using qxyz.

                Variables are:
                        xyz magnetic vector index (ie x=0 etc)

                        mr long  vectors of m data points
                                rbeta   -       Householder intermediate value
                                Qphi    -       Q x phi = y bar

                        mr x 3 array of m data point sets
                                B [x, y and z]- measured/true surface mag field
                                Bv      -       volume correction (not used)

                        6 element vectors       [0],[1] and [2] are quadrupole [3],[4] and [5] are dipole
                                a       -       source coordinates used to generate the data
                                aest    -       estimated increment of source coordinates


                        nc element vector [0] to [8] = quadrupole  [9] to [11] = dipole
                                da      -       quadrupole vector strengths used to generate the data
                                daest   -       estimated quadrupole vector strengths
                                perm    -       permutation vector set up in the householder decomposition.
                                                sbreak indicates pseudo rank deficiency point as controlled
                                                        by precision.

                        rank x rank matrix
                                T11     -       orthog transformed upper triang. or its inverse.

                        rank x (mr-rank) matrix
                                T11Q    -       orthogonal complement projector for the source
                                                        position increments

                        mr x rank matrices ('rank'= 12 = nc linear functionals
```

```
                                but rank changes to 6 when the x,y,z coordinates are estimated for each pole)
                        A       -       mr surface coordinates

                mr x nc matrices ('rank'= 12 = nc linear functionals)
                        fv      -       mr sets nc x 6 non-linear functionals - ie nc for each of Bx, By and Bz
                                                The surface mag field due to each multipole.
                        v       -       mr sets non-linear functionals from one vector (ie Bx or By or Bz)
                        dvda    -       mr sets of partial differentials of v wrt to each a.
                        vtv     -       basis columns for v from Householder

                mr x mr matrix
                        Q       -       orthogonal projector

                mr x k x k matices where k is the multipole order
                        BxA, ByA, BzA - field point constants from the surface integration (A even series)
                        BxB, ByB, BzB - (B odd series)

                qxyz            -       are the least linearly dependent columns for the linear projection
*/

mrmat=mr-1;
ncmat=0;
rankmat=rank-1;
//b is set up as one column array to allow gaussj2 to work
b=make2darray(&rankmat,&ncmat);


ncmat=nc-1;
dvda=make2darray(&mrmat,&ncmat);
Q=make2darray(&mrmat,&mrmat);
Qphi=malloc((mr)*sizeof(float));
T11=make2darray(&rankmat,&rankmat);

mrmat += (12-rank);
ncmat = 17;

v=make2darray(&mrmat,&ncmat);
vtv=make2darray(&mrmat,&ncmat);
fv=make2darray(&mrmat,&ncmat);

ncmat=mr-rank-1;
T11Q=make2darray(&rankmat,&ncmat);

perm=malloc((nc)*sizeof(int));
wd=malloc((nc)*sizeof(float));
rbeta=malloc((nc)*sizeof(float));


//Compute the magnetic field due to the volume currents from the quadrupole source (3b) and the dipole source (3d).
////These are placed in v[i][0 to 8] and v[i][9,10 and11] respectively
fharmmcg3b(0,a,fv,BcA,BcB,da,start,end,bc,mr);
fharmmcg3d(0,a+3,fv,BcA,BcB,da,start,end,bc,mr);

for (i=0;i<mr;i++) {

        Bnew[i] = 0;
        for (j=0;j<12;j++) v[i][j]=0;

        /*** The volume current terms  ***/
        //Need the negative functionals from the dipole and quadrupole volume models
        //Dipole terms are at positions [9], [10] and [11] in the matrix of functionals
        //cf. B = Bd - Bv

        for (j=0;j<12;j++) {
                        v[i][qxyzv[j]] = -fv[i][qxyzv[j]];
        }

        //compute Bnew for all elements of the quadrupole
        for (j=0;j<9;j++) {
                Bnew[i] -=da[j]*fv[i][j];
        }

        //compute inf medium mag field functionals for the dipole
        //Note that these are added to positions [9] [10] and [11]
        fdipmcg(*(A+i),a+3,*(fv+i));

        // Dipole strengths are held in da[ ]
```

```
		if (flagxyz==0)
		{

			v[i][10] +=fv[i][0];
			v[i][11] +=fv[i][1];
		}

		else if (flagxyz==1)
		{
			v[i][9] +=fv[i][3];
			v[i][11] +=fv[i][2];
		}

		else{
			v[i][9] +=fv[i][4];
			v[i][10] +=fv[i][5];
		}

		//Now compute Bnew using total functionals from the dipole
		Bnew[i] += (da[9]*v[i][9] + da[10]*v[i][10] + da[11]*v[i][11]);

		/*** The quadrupole terms  ***/
		//compute inf medium mag field functionals for the quadrupole terms

		fquad1mcg(*(A+i),a,*(fv+i));

		// Quadrupole strengths are held in da[0 - 8] indexed by qxyz


		for (j=0;j<5;j++) {
				v[i][qxyz[j]] +=fv[i][j+5*flagxyz];
				Bnew[i] +=da[qxyz[j]]*fv[i][j+5*flagxyz];
			}
		//Adjust Bnew for yy term
		if (flagxyz==0) Bnew[i] +=  2*da[4]*fv[i][flagxyz*5];
		if (flagxyz==1) Bnew[i] += 0.5*da[4]*fv[i][flagxyz*5];
		if (flagxyz==2) Bnew[i] -=    da[4]*fv[i][flagxyz*5];


	}

/*	for (i=0;i<mr;i++) {
			for (j=0;j<rank;j++)
			{
				fprintf(ifp9,"%15.4f  ",v[i][j]);
			}
			fprintf(ifp9,"\n");
		}
	fprintf(ifp9,"\n\n\n");
*/
	nc=12; rank=10;
	househ (v,&mr,&nc,&rank,perm,vtv,&precision,&sbreak,rbeta);

	for (i=0;i<mr;i++)
		{
			Qphi[i]=B[i];
		}

	startrow=0;
	for (j=0;j<=sbreak;j++) {
		w=0.0;
		for (i=startrow;i<mr;i++) {
			w += vtv[i][perm[j]]*Qphi[i];
		}
		w=w*rbeta[perm[j]];
		for (i=startrow;i<mr;i++) {
			Qphi[i] +=w*vtv[i][perm[j]];
		}
		if (startrow<(rank-1)) startrow++;
	}


/*	for (i=0;i<mr;i++)
	{
		for (j=0;j<nc;j++) fprintf(ifp9,"%15.2f  ",v[i][j]);
		fprintf(ifp9,"\n");
	}
	fprintf(ifp9,"\n\n");
```

```
        for (i=0;i<mr;i++)
        {
                fprintf(ifp9," %15.4f",Qphi[i]);
                fprintf(ifp9,"\n");
        }

        fprintf(ifp9,"\n\n");
*/
        //Create T11 and then inverse

        for (i=0;i<rank;i++) {
                for (j=0;j<rank;j++) T11[i][j]=v[i][perm[j]];
        }

/*      fprintf(ifp9,"\n");
        for (i=0;i<rank;i++) fprintf(ifp9,"%15.4f  ",rbeta[perm[i]]);
        {
                for (j=0;j<rank;j++) fprintf(ifp9,"%15.4f  ",T11[i][j]);
                fprintf(ifp9,"\n");
        }
        fprintf(ifp9,"\n\n\n");
*/
        gaussj2(T11,rank,b,1);

/*      for (i=0;i<rank;i++)
        {
                for (j=0;j<rank;j++) fprintf(ifp9,"%15.4f  ",T11[i][j]);
                fprintf(ifp9,"\n");
        }
*/
        //Now compute and display quadrupole strengths re-permutated back


        for (i=0;i<rank;i++) {
                daest[perm[i]]=0.0;
                for (j=0;j<rank;j++){
                        daest[perm[i]] +=T11[i][j]*Qphi[j];
                }
        }

        //fprintf(ifp9,"Dx=%f  Dy=%f  Dz=%f \nQxx=%f  Qxy=%f  Qzy=%f  Qxz=%f
//Qyz=%f\n",daest[9],daest[10],daest[11],daest[qxyz[0]],daest[qxyz[1]],daest[qxyz[2]],daest[qxyz[3]].daest[qxyz[4]]);
        //fprintf(ifp9,"\n");
        //fclose(ifp9);
        /* Now find the x, y and z increments */

        //The estimated Quadrupole strengths x the partial derivs wrt each non-lin var.
        //These are summed wrt d/dx  d/dy and d/dz respectively to give an mr x 3 matrix

        kay=6; //for the tensor arithmetic

        /*** Quadrupole vol current contribution to the derivatives in v[0,1,2] ***/
        fharmmcg3b(1,a,v,BcA,BcB,da,start,end,bc,mr);


        /*** Quadrupole inf medium contribution to the derivatives ***/

        for (i=0;i<mr;i++) {

                //diffs wrt to quadrupole location are added to [0] [1] and [2].
                fquad0mcg(flagxyz,*(A+i),a,*(dvda+i),da);
                for (j=0;j<3;j++) {
                        v[i][j] -= dvda[i][j]; //Require the negative orthog dvda
                }
        }

        /*** Add the Dipole contribution ***/

        //Derivative contribution due to the volume currents
        //Diffs wrt to dipole location are in [3] [4] and [5]
        fharmmcg3d(1,a+3,v,BcA,BcB,da,start,end,bc,mr);

        //compute the differentials from the inf medium dipole model and add their negative
        //to positions [3.4.5]
        for (i=0;i<mr;i++)
        {
                fdip0mcg(flagxyz,*(A+i),a+3,*(dvda+i),&da[9]);
                for (j=0;j<3;j++) {
```

```
                        v[i][j+3] -= dvda[i][j]; //Require the negative orthog dvda
                }
        }

/*      printf("\n");
        for (i=0;i<11;i++)
        {
                for (j=0;j<kay;j++) printf("%8.4f ",v[i][j]);
                printf("\n");
        }
*/


        //These are multiplied by the negative orthog matrix but starting
        // from (rankmat + 1) to form a new matrix v

        startrow=0;
        for (j=0;j<=sbreak;j++) {
                for(k=0;k<kay;k++) {
                        wd[k]=0.0;
                        for (i=startrow;i<mr;i++) {
                                wd[k] += vtv[i][perm[j]]*v[i][k];
                        }
                        wd[k]=wd[k]*rbeta[perm[j]];
                        for (i=startrow;i<mr;i++) {
                                v[i][k] +=wd[k]*vtv[i][perm[j]];
                        }
                }
                if (startrow<(rank-1)) startrow++;
        }

        //Take the mr-rank rows
        for (i=rank;i<mr;i++) {
                for (j=0;j<kay;j++) {
                        v[i-rank][j]=v[i][j];
                }
        }
        //Set the remainder of v to zero
        for (i=mr-rank;i<mr;i++)
                for (j=0;j<kay;j++) v[i][j]=0;


        //Now add upper triang Cholesky factor to the end of v. This version
        // asks for a control factor which is used as a multiplier on an identity matrix
        for (i=mr-kay;i<mr;i++) {
                for (j=0;j<kay;j++) {
                        if ((i-(mr-kay)) == j) v[i][j] = vj;
                        else v[i][j] = 0;
                }
        }

/*      for (i=0;i<11;i++)
        {
                for (j=0;j<kay;j++) printf("%8.4f ",v[i][j]);
                printf("\n");
        }
*/
        //Change rank to 6
        nc=6;
        househ (v,&mr,&nc,&kay,perm,vtv,&precision,&sbreak,rbeta);

/*      printf("\n");
        for (i=0;i<11;i++)
        {
                for (j=0;j<nc;j++) printf("%8.4f ",v[i][j]);
                printf("\n");
        }
*/


        //Remove the first rank elements of Qphi and add rank zeros
        for (i=0;i<mr;i++) {
                if (i<(mr-rank)) Qphi[i]=Qphi[i+rank];
                else Qphi[i]=0;
        }
        //Now orthogonalize
        startrow=0;
        for (j=0;j<=sbreak;j++) {
```

```
                w=0.0;
                for (i=startrow;i<mr;i++) {
                        w += vtv[i][perm[j]]*Qphi[i];
                }
                w=w*rbeta[perm[j]];
                for (i=startrow;i<mr;i++) {
                        Qphi[i] +=w*vtv[i][perm[j]];
                }
                if (startrow<(kay-1)) startrow++;
        }

/*      printf("\nQphi");
        for (i=0;i<mr;i++) printf("%f\n",Qphi[i]);
*/

        //Create T11 and then inverse

        for (i=0;i<kay;i++) {
                for (j=0;j<kay;j++) T11[i][j]=v[i][perm[j]];
        }

/*      printf("\n");
        for (i=0;i<kay;i++)
        {
                for (j=0;j<kay;j++) printf("%8.4f ",T11[i][j]);
                printf("\n");
        }
*/
        gaussj2(T11,kay,b,1);

/*      for (i=0;i<kay;i++)
        {
                for (j=0;j<kay;j++) printf("%8.4f ",T11[i][j]);
                printf("\n");
        }
*/
        //Multiply new Q by T11 inverse. Only the first kay rows of Q are
        //needed since T11 is a kay x kay matrix. T11Q is also multiplied
        // by the permutation matrix.
/**
        for (i=0;i<kay;i++){
                for (j=0;j<(mr-kay);j++) {
                        T11Q[perm[i]][j]=0;
                        for (k=0;k<kay;k++) T11Q[perm[i]][j] += T11[i][k]*Q[k][j];
                }
        }
**/
/*      for (i=0;i<kay;i++)
        {
                for (j=0;j<(mr-kay);j++) printf("%8.4f ",T11Q[i][j]);
                printf("\n");
        }
*/

/**
        for (i=0;i<kay;i++){
                aest[i]=0;
                for (j=0;j<(mr-kay);j++) {
                        aest[i] += T11Q[i][j]*Qphi[j+kay];
                }
        }
**/
        for (i=0;i<kay;i++){
                aest[perm[i]]=0;
                for (j=0;j<(kay);j++) {
                        aest[perm[i]] += T11[i][j]*Qphi[j];
                }
        }

        //printf("\n\ndx=%8.5f dy=%8.5f dz=%8.5f",aest[0],aest[1],aest[2]);
        //printf("\nqx=%8.5f qy=%8.5f qz=%8.5f",aest[3],aest[4],aest[5]);

        free2darray(v,&mrmat);
        free2darray(fv,&mrmat);
        free2darray(b,&rankmat);
        free2darray(dvda,&mrmat);
        free(Qphi);
        free2darray(T11,&rankmat);
```

```
        free2darray(Q,&mrmat);
        free2darray(vtv,&mrmat);
        free2darray(T11Q,&rankmat);
        free(perm);
        free(wd);
        free(rbeta);

        return;
}
```

```
/**** fquad4ecg
        Dipole + Quadrupole electric potential functionals
        and first differentials ****/

#include <math.h>
#include <stdio.h>
float Fxxx (float,float,float,float,float,float);
float Fxxy (float,float,float,float,float,float,float,float);
float Fxyz (float,float,float,float,float,float,float,float,float);
float Fxyx (float,float,float,float,float,float,float,float);
float Fxyy (float,float,float,float,float,float,float);
float rho,rho2,rho3,rho4,rho5,rho7,gma,rgma,rgma2,rgma3;

void fquad4ecg(int flag,float x[],float a[],float *v,float dvda[],int na,float *Q)
{
        /****************** Forward ECG potential v on the surface of a sphere from
                                an internal arbitrarily located dipole and quadrupole. Sphere radius = 1
                                Function also computes the first partial differential of
                                v wrt to the quadrupole coordinates a[i].          ******************/
        //Combination of for3ecg and fquad2ecg for combined dipole and quadrupole functionals
//Separate locations
//This model only uses the 5 linearly independent quadrupole components
        //which are passed in Q[0..4]
        //Computing data for Fxx, Fxy, Fxz, Fyy and Fyz

        int     i,j,k,pos=0,idx=0;
        float   xdiff[3],xsq[3],xgma[3],delx[3],xrho[3],xidiff,xjdiff,old;

/*** First do the dipole terms ***/
        //Define local and global variables
        gma=0;
        rho2=0;
        for (i=0;i<3;i++) {
                gma += x[i]*a[i];
                xdiff[i]=x[i]-a[i];
                rho2 += xdiff[i]*xdiff[i];
                xsq[i]=x[i]*x[i]-1;
        }

        rho=pow(rho2,0.5);
        rho4=pow(rho2,2);
        rho3=pow(rho,3);
        rho5=pow(rho,5);
        rho7=pow(rho,7);
        rgma=rho+1-gma;
        rgma2=pow(rgma,2);
        rgma3=pow(rgma,3);

        for (i=0;i<3;i++) {
                xgma[i]=x[i]*gma-a[i];
                delx[i]=(xdiff[i]/rho)+x[i];
                xrho[i]=x[i]*rho+xdiff[i];
        }
        if (flag) {

                //Compute coefficients for Dx, Dy and Dz and place at positions v[5], [6] and [7]
                for (i=0;i<3;i++) {
                *(v+i+5) = 2*(x[i]-a[i])/rho3 + x[i]/rho + (x[i]*gma-a[i])/(rho*rgma);
                }

        }

        else {
                //Compute the dipole partial differential tensor multiplied by the dipole strengths which are held
                // in Q[5,6 and 7]
                for (i=0;i<3;i++) {
                        dvda[i]=0;
                        for(j=0;j<3;j++) {
                                xidiff=x[i]-a[i]; xjdiff=x[j]-a[j];
                                dvda[i] += Q[j+5]*((i==j)*(-2)/rho3
                                        + 6*xidiff*xjdiff/rho5
                                        + x[i]*xjdiff/rho3
                                        + (x[i]*x[j] - (i==j))/(rho*rgma)
                                        + ((x[i]*gma)-a[i])*(xjdiff*(rho+rgma) + rho2*x[j])/(rho3*rgma*rgma));
                        }
                }
        }
```

```
/*** Now do the quadrupole terms ***/

        gma=0;
        rho2=0;
        for (i=0;i<3;i++) {
                gma += x[i]*a[i+3];
                xdiff[i]=x[i]-a[i+3];
                rho2 += xdiff[i]*xdiff[i];
                xsq[i]=x[i]*x[i]-1;
        }

        rho=pow(rho2,0.5);
        rho4=pow(rho2,2);
        rho3=pow(rho,3);
        rho5=pow(rho,5);
        rho7=pow(rho,7);
        rgma=rho+1-gma;
        rgma2=pow(rgma,2);
        rgma3=pow(rgma,3);

        for (i=0;i<3;i++) {
                xgma[i]=x[i]*gma-a[i+3];
                delx[i]=(xdiff[i]/rho)+x[i];
                xrho[i]=x[i]*rho+xdiff[i];
        }


        if (flag) {
                //Compute compute coefficients for Qxx, Qxy, Qxz, Qyy and Qyz
                for (i=0;i<2;i++) {
                        for(j=0;j<3;j++) {
                                if (i<=j) {
                                        xidiff=x[i]-a[i+3]; xjdiff=x[j]-a[j+3];
                                        *(v+pos) = (i==j)*(-2)/rho3
                                                + 6*xidiff*xjdiff/rho5
                                                + x[i]*xjdiff/rho3
                                                + (x[i]*x[j] - (i==j))/(rho*rgma)
                                                + ((x[i]*gma)-a[i+3])*(xjdiff*(rho+rgma) + rho2*x[j])/(rho3*rgma*rgma);
                                        pos++;
                                }
                        }
                }
        }

        else {
                //Compute the quadrupole partial differential tensor multiplied by the quadrupole strengths
                //each dvda[ij] for x(j)=(x,y,z) and for x(i)=(x,y,z) is
                //the summation of d{d(Fx(i))/dx(j)}/dx(k)  for x(k)=(x,y,z)
                //resulting in 3 terms held in dvda[3,4 and 5]

                pos=3;
                for (k=0;k<3;k++) {
                        old=0;
                        idx=0;
                        dvda[pos]=0;
                        for (j=0;j<2;j++) {
                            for (i=j;i<3;i++) {
                                if ((i==j)&&(i==k))      dvda[pos] +=Q[idx]*Fxxx(xdiff[i],x[i],xrho[i],xsq[i],delx[i],xgma[i]);
                                else if((i==j)&&(i!=k))  dvda[pos]+=Q[idx]*Fxxy(xdiff[i],xdiff[k],x[i],xsq[i],delx[k],delx[i],x[k],xgma[i]);
                                else if((i!=j)&&(j!=k)&&(i!=k)) dvda[pos]+=Q[idx]*Fxyz(xdiff[i],xdiff[j],xdiff[k],x[i],x[j],x[k],delx[k],delx[j],xgma[i]);
                                else if ((i!=j)&&(i==k))  dvda[pos] += Q[idx]*Fxyx(xdiff[j],xdiff[i],x[i],delx[i],x[j],xsq[i],delx[j],xgma[i]);
                                else if((i!=j)&&(j==k))   dvda[pos] += Q[idx]*Fxyy(xdiff[i],xdiff[j],x[i],xrho[i],delx[j],x[j],xgma[i]);
                                idx++;
                                //printf("\n%f",dvda[pos]-old);
                                //printf("\n");
                                old=dvda[pos];
                            }
                        }
                        pos++;
                }
        }

        return;
}
```

```
/**** fharmmcg3d
        Magnetic field from Dipole volume currents****/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "nrutil.h"


float fact(int);


/***** Dipole version ****/

void fharmmcg3d(int ind,float a[],float **Bv,float ***BcA,float ***BcB,float dvda[],int start,int end,float **bc,int mr)
{
        /****************** Forward MCG field Bv(xyz) on the surface of a sphere due to
                                                        volume current sources from an internal arbitrarily located dipole.
                                                        Sphere radius = 1
                                                        Harmonic series version.

                                                        This version also computes the functional differentials ******************/

        //Note that the dipole strengths are held in dvda[nc] and the potential
        //is calculated from Dx,Dy,Dz held in  dvda[9,10,11]
        //Bv is now Bv[i][x,y or z]

        /*** How the differentials are built up.

                        For each term there is an [n][m] and an even (A) and an odd (B) series:

                        Fr, Fth and Fph are d/dr d/dth and d/dph of F {eg A seires FA[0][n][m],FA[1][n][m],FA[2][n][m]}
                        Frr to Fphph are d/dr to d/dph of Fr to Fph {eg A seires FrA[0][0][n][m] thru FrA[2][2][n][m]}
                        Frrr to Fphphph are d/dr d/dth and d/dph of Frr to Fphph {generated by fnc octapole}

                        Fx, Fy and Fz are the cartesian assembly of Fr, Fth and Fph

                        Fxr to Fzph are d/dr to d/dph of Fx to Fz
                        Fxx to Fzz are the cartesian assembly of Fxr to Fzph {eg A series Fx[0][0][n][m] thru Fx[2][2][n][m]}

                        Fxrr to Fzphph are d/dr to d/dph of Fxr to Fzph
                        Fxxr to Fzzph are d/dr to d/dph of Fxx to Fzz
                        Fxxx to Fzzz are the cartesian assembly of Fxxr to Fzzph

                        The final series can be combined with [n][m]  A and B  surface constant terms BcA and BcB  ***/


        int             i,j,k,n,m=3,nmat,mmat;
        float           r,asph[3],Sths,Cths,*Sphs,*Cphs,*rn1,*rn2,*rn3,**Ps,**LmbdA,**LmbdB,ctcp,stsp,ctsp,stcp,
                        **Pdiff1,**Pdiff2,**Pdiff3,***FA,***FB,
                        ****FxA,****FrA,
                        ****FxB,****FrB,

                        **FxrA, **FxthA, **FxphA,**FyrA, **FythA, **FyphA, **FzrA, **FzthA,**FzphA,
                        **FxrB, **FxthB, **FxphB,**FyrB, **FythB, **FyphB, **FzrB, **FzthB,**FzphB;


        /******* The multipole order is k ************/


        Sphs=malloc((end+1)*sizeof(float));

        Cphs=malloc((end+1)*sizeof(float));

        rn1=malloc((end+1)*sizeof(float));
        rn2=malloc((end+1)*sizeof(float));
        rn3=malloc((end+1)*sizeof(float));

        nmat=end+3;
        mmat=end+3;
        Ps=make2darray(&nmat,&mmat);
        Pdiff1=make2darray(&nmat,&mmat);

        Pdiff2=make2darray(&nmat,&mmat);
        Pdiff3=make2darray(&nmat,&mmat);

        nmat=end;
```

```
mmat=end;
m=3;

FA=make3darray(&m,&nmat,&mmat);

FrA=make4darray(&m,&m,&nmat,&mmat);
FxA=make4darray(&m,&m,&nmat,&mmat);

FxrA=make2darray(&nmat,&mmat);
FxthA=make2darray(&nmat,&mmat);
FxphA=make2darray(&nmat,&mmat);
FyrA=make2darray(&nmat,&mmat);
FythA=make2darray(&nmat,&mmat);
FyphA=make2darray(&nmat,&mmat);
FzrA=make2darray(&nmat,&mmat);
FzthA=make2darray(&nmat,&mmat);
FzphA=make2darray(&nmat,&mmat);

FB=make3darray(&m,&nmat,&mmat);

FrB=make4darray(&m,&m,&nmat,&mmat);
FxB=make4darray(&m,&m,&nmat,&mmat);

FxrB=make2darray(&nmat,&mmat);
FxthB=make2darray(&nmat,&mmat);
FxphB=make2darray(&nmat,&mmat);
FyrB=make2darray(&nmat,&mmat);
FythB=make2darray(&nmat,&mmat);
FyphB=make2darray(&nmat,&mmat);
FzrB=make2darray(&nmat,&mmat);
FzthB=make2darray(&nmat,&mmat);
FzphB=make2darray(&nmat,&mmat);

//Ensure Bv is zero
if(ind==0){
        for (i=0;i<mr;i++) {
                for (j=9;j<12;j++) Bv[i][j]= 0;
                }
}
else{
        for (i=0;i<mr;i++) {
                for (j=3;j<6;j++) Bv[i][j]= 0;
                }

}


//convert to spherical
//Note the a values are constant and should be done in main

xyztosph(a,asph);
r=asph[0];


Sths=sin(asph[1]);

Cths=cos(asph[1]);
for (i=0;i<=end;i++) {

 Sphs[i]=sin(i*asph[2]);

 Cphs[i]=cos(i*asph[2]);

 rn1[i]=pow(r,(i-1));
 rn2[i]=pow(r,(i-2));
 rn3[i]=pow(r,(i-3));
}

nmat=end+3;
mmat=end+3;
lgnd(Ps,Cths,nmat);

stsp=Sths*Sphs[1];
ctcp=Cths*Cphs[1];
stcp=Sths*Cphs[1];
ctsp=Cths*Sphs[1];
```

```
//Compute Legendre differentials

/**** New version - allows for Sths = 0 ***/
 for (n=0;n<=(end);n++){
                        //printf("\nn= %d ",n);
              for (m=0;m<=n;m++){
                        if (m==0) Pdiff1[n][m] = -Ps[n][1];
                        else if (n<=(end+2)) {
                                        Pdiff1[n][m]=((n+m)*(n-m+1)*Ps[n][m-1])/2;
                                        if (m<n) Pdiff1[n][m] -= (Ps[n][m+1])/2;
                                }
                                if (m==0) Pdiff2[n][m] = (Ps[n][2] -n*(n+1)*Ps[n][0])/2;
                                else if (m==1) Pdiff2[n][m] = -((3*n*n+3*n-2)*Ps[n][1] - Ps[n][3])/4;
                                else Pdiff2[n][m] = (((n+m-1)*(n-m+2)*Ps[n][m-2] - Ps[n][m])*(n+m)*(n-m+1) - (n+m+1)*(n-m)*Ps[n][m] - Ps[n][m+2])/4;

                                if (m==0) Pdiff3[n][m] = ((3*n*n+3*n-2)*Ps[n][1] - Ps[n][3])/4;
                                else if (m==1) Pdiff3[n][m] = -(3*n*n+3*n-2)*n*(n+1)*Ps[n][0]/8 + (n+2)*(n-1)*Ps[n][3]/2 -Ps[n][4]/8;
                                else if (m==2) Pdiff3[n][m] = ((3*n*n+3*n+20)*Ps[n][3] - Ps[n][5] - 4*(n+2)*(n-1)*(n+2)*(n-1)*Ps[n][1])/8;
                                else Pdiff3[n][m] = ((n+m-1)*(n-m+2)*(n+m-2)*(n-m+3)*Ps[n][m-3] - (3*n*n+3*n-3*m*m+3*m-2)*Ps[n][m-1])*(n+m)*(n-m+1)/8
                                                                                + ((3*n*n+3*n-3*m*m-3*m-2)*Ps[n][m+1])/8;


                        }
                       }


/**** Old version ******
              for (n=0;n<=end;n++){
                        printf("\nn= %d ",n);
                for (m=0;m<=n;m++){
                        if (n<=(end+2)) {
                                        Pdiff1[n][m]=-(n+1)*Ps[n][m]*Cths/Sths + (n-m+1)*Ps[n+1][m]/Sths;
                                }

                        if ((n>1)&&(n<=(end+1))) Pdiff2[n-1][m]= (-(n-m)*Ps[n][m]+(n-m)*Pdiff1[n][m]*Sths/Cths +
                                                                n*Ps[n-1][m]/Cths - n*Pdiff1[n-1][m]*Sths)*Cths/(Sths*Sths);
                        if ((n>2)&&(n<=(end+2))) Pdiff3[n-2][m]= (n-1-m)*(Pdiff2[n-1][m] - 2*Pdiff1[n-1][m]*Cths/Sths +
                                                                Ps[n-1][m]*(1+2*Cths*Cths/(Sths*Sths)))/Sths +
                                                                (n-1)*(-2*Ps[n-2][m]*Cths/Sths+2*Pdiff1[n-2][m]-Pdiff2[n-2][m]*Sths*Cths)/(Sths*Sths);
                        printf("%8.1f ",Pdiff2[n][m]);

                }
               }

              printf("\n");***********/

/*** Sths must not be zero in this version***/

              for (n=start;n<=end;n++){
                for (m=0;m<=n;m++){
                        FA[0][n][m]= n*rn1[n]*Ps[n][m];
                        FA[1][n][m]=rn1[n]*Pdiff1[n][m];
                        FA[2][n][m]=m*rn1[n]*Ps[n][m]/Sths;

                        FB[0][n][m] = FA[0][n][m]*Sphs[m];
                        FA[0][n][m] *= Cphs[m];

                        FB[1][n][m] = FA[1][n][m]*Sphs[m];
                        FA[1][n][m] *= Cphs[m];

                        FB[2][n][m] = FA[2][n][m]*Cphs[m];
                        FA[2][n][m] *= -Sphs[m];

                        FrA[0][0][n][m]=n*(n-1)*rn2[n]*Ps[n][m];
                        FrA[0][1][n][m]=n*rn2[n]*Pdiff1[n][m];
                        FrA[0][2][n][m]=n*m*rn2[n]*Ps[n][m]/Sths;

                        FrA[1][0][n][m]=(n-1)*rn2[n]*Pdiff1[n][m];
                        FrA[1][1][n][m]=rn2[n]*Pdiff2[n][m];
                        FrA[1][2][n][m]=m*rn2[n]*Pdiff1[n][m]/Sths;

                        FrA[2][0][n][m]=(n-1)*m*rn2[n]*Ps[n][m]/Sths;
                        FrA[2][1][n][m]=m*rn2[n]*(-Cths*Ps[n][m]/Sths+Pdiff1[n][m])/Sths;
                        FrA[2][2][n][m]=-m*m*rn2[n]*Ps[n][m]/(Sths*Sths);

                        FrB[0][0][n][m]= FrA[0][0][n][m]*Sphs[m];
                        FrB[0][1][n][m]= FrA[0][1][n][m]*Sphs[m];
```

```
FrB[0][2][n][m]= FrA[0][2][n][m]*Cphs[m];

FrB[1][0][n][m]= FrA[1][0][n][m]*Sphs[m];
FrB[1][1][n][m]= FrA[1][1][n][m]*Sphs[m];
FrB[1][2][n][m]= FrA[1][2][n][m]*Cphs[m];

FrB[2][0][n][m]= FrA[2][0][n][m]*Cphs[m];
FrB[2][1][n][m]= FrA[2][1][n][m]*Cphs[m];
FrB[2][2][n][m]= FrA[2][2][n][m]*Sphs[m];

FrA[0][0][n][m] *= Cphs[m];
FrA[0][1][n][m] *= Cphs[m];
FrA[0][2][n][m] *= -Sphs[m];

FrA[1][0][n][m] *= Cphs[m];
FrA[1][1][n][m] *= Cphs[m];
FrA[1][2][n][m] *= -Sphs[m];

FrA[2][0][n][m] *= -Sphs[m];
FrA[2][1][n][m] *= -Sphs[m];
FrA[2][2][n][m] *= Cphs[m];

FxrA[n][m] = FrA[0][0][n][m]*Sths*Cphs[1] + FrA[1][0][n][m]*Cths*Cphs[1] - FrA[2][0][n][m]*Sphs[1];
FyrA[n][m] = FrA[0][0][n][m]*Sths*Sphs[1] + FrA[1][0][n][m]*Cths*Sphs[1] + FrA[2][0][n][m]*Cphs[1];
FzrA[n][m] = FrA[0][0][n][m]*Cths - FrA[1][0][n][m]*Sths;

FxthA[n][m] = (FrA[0][1][n][m] - FA[1][n][m]/r)*Sths*Cphs[1] + (FrA[1][1][n][m] + FA[0][n][m]/r)*Cths*Cphs[1] - FrA[2][1][n][m]*Sphs[1];
FythA[n][m] = (FrA[0][1][n][m] - FA[1][n][m]/r)*Sths*Sphs[1] + (FrA[1][1][n][m] + FA[0][n][m]/r)*Cths*Sphs[1] + FrA[2][1][n][m]*Cphs[1];
FzthA[n][m] = (FrA[0][1][n][m] - FA[1][n][m]/r)*Cths - (FrA[1][1][n][m] + FA[0][n][m]/r)*Sths;

FxphA[n][m] = FrA[0][2][n][m]*Sths*Cphs[1] + FrA[1][2][n][m]*Cths*Cphs[1] - FrA[2][2][n][m]*Sphs[1] -
                    FA[0][n][m]*Sphs[1]/r - FA[1][n][m]*Cths*Sphs[1]/(r*Sths) - FA[2][n][m]*Cphs[1]/(r*Sths);
FyphA[n][m] = FrA[0][2][n][m]*Sths*Sphs[1] + FrA[1][2][n][m]*Cths*Sphs[1] + FrA[2][2][n][m]*Cphs[1] +
                    FA[0][n][m]*Cphs[1]/r + FA[1][n][m]*Cths*Cphs[1]/(r*Sths) - FA[2][n][m]*Sphs[1]/(r*Sths);
FzphA[n][m] = FrA[0][2][n][m]*Cths - FrA[1][2][n][m]*Sths;


FxrB[n][m] = FrB[0][0][n][m]*Sths*Cphs[1] + FrB[1][0][n][m]*Cths*Cphs[1] - FrB[2][0][n][m]*Sphs[1];
FyrB[n][m] = FrB[0][0][n][m]*Sths*Sphs[1] + FrB[1][0][n][m]*Cths*Sphs[1] + FrB[2][0][n][m]*Cphs[1];
FzrB[n][m] = FrB[0][0][n][m]*Cths - FrB[1][0][n][m]*Sths;

FxthB[n][m] = (FrB[0][1][n][m] - FB[1][n][m]/r)*Sths*Cphs[1] + (FrB[1][1][n][m] + FB[0][n][m]/r)*Cths*Cphs[1] - FrB[2][1][n][m]*Sphs[1];
FythB[n][m] = (FrB[0][1][n][m] - FB[1][n][m]/r)*Sths*Sphs[1] + (FrB[1][1][n][m] + FB[0][n][m]/r)*Cths*Sphs[1] + FrB[2][1][n][m]*Cphs[1];
FzthB[n][m] = (FrB[0][1][n][m] - FB[1][n][m]/r)*Cths - (FrB[1][1][n][m] + FB[0][n][m]/r)*Sths;

FxphB[n][m] = FrB[0][2][n][m]*Sths*Cphs[1] + FrB[1][2][n][m]*Cths*Cphs[1] - FrB[2][2][n][m]*Sphs[1] -
                    FB[0][n][m]*Sphs[1]/r - FB[1][n][m]*Cths*Sphs[1]/(r*Sths) - FB[2][n][m]*Cphs[1]/(r*Sths);
FyphB[n][m] = FrB[0][2][n][m]*Sths*Sphs[1] + FrB[1][2][n][m]*Cths*Sphs[1] + FrB[2][2][n][m]*Cphs[1] +
                    FB[0][n][m]*Cphs[1]/r + FB[1][n][m]*Cths*Cphs[1]/(r*Sths) - FB[2][n][m]*Sphs[1]/(r*Sths);;
FzphB[n][m] = FrB[0][2][n][m]*Cths - FrB[1][2][n][m]*Sths;


FxA[0][0][n][m] = FxrA[n][m]*Sths*Cphs[1] + FxthA[n][m]*Cths*Cphs[1] - FxphA[n][m]*Sphs[1];
FxA[1][0][n][m] = FxrA[n][m]*Sths*Sphs[1] + FxthA[n][m]*Cths*Sphs[1] + FxphA[n][m]*Cphs[1];
FxA[2][0][n][m] = FxrA[n][m]*Cths - FxthA[n][m]*Sths;

FxA[0][1][n][m] = FyrA[n][m]*Sths*Cphs[1] + FythA[n][m]*Cths*Cphs[1] - FyphA[n][m]*Sphs[1];
FxA[1][1][n][m] = FyrA[n][m]*Sths*Sphs[1] + FythA[n][m]*Cths*Sphs[1] + FyphA[n][m]*Cphs[1];
FxA[2][1][n][m] = FyrA[n][m]*Cths - FythA[n][m]*Sths;

FxA[0][2][n][m] = FzrA[n][m]*Sths*Cphs[1] + FzthA[n][m]*Cths*Cphs[1] - FzphA[n][m]*Sphs[1];
FxA[1][2][n][m] = FzrA[n][m]*Sths*Sphs[1] + FzthA[n][m]*Cths*Sphs[1] + FzphA[n][m]*Cphs[1];
FxA[2][2][n][m] = FzrA[n][m]*Cths - FzthA[n][m]*Sths;

FxB[0][0][n][m] = FxrB[n][m]*Sths*Cphs[1] + FxthB[n][m]*Cths*Cphs[1] - FxphB[n][m]*Sphs[1];
FxB[1][0][n][m] = FxrB[n][m]*Sths*Sphs[1] + FxthB[n][m]*Cths*Sphs[1] + FxphB[n][m]*Cphs[1];
FxB[2][0][n][m] = FxrB[n][m]*Cths - FxthB[n][m]*Sths;

FxB[0][1][n][m] = FyrB[n][m]*Sths*Cphs[1] + FythB[n][m]*Cths*Cphs[1] - FyphB[n][m]*Sphs[1];
FxB[1][1][n][m] = FyrB[n][m]*Sths*Sphs[1] + FythB[n][m]*Cths*Sphs[1] + FyphB[n][m]*Cphs[1];
FxB[2][1][n][m] = FyrB[n][m]*Cths - FythB[n][m]*Sths;

FxB[0][2][n][m] = FzrB[n][m]*Sths*Cphs[1] + FzthB[n][m]*Cths*Cphs[1] - FzphB[n][m]*Sphs[1];
FxB[1][2][n][m] = FzrB[n][m]*Sths*Sphs[1] + FzthB[n][m]*Cths*Sphs[1] + FzphB[n][m]*Cphs[1];
FxB[2][2][n][m] = FzrB[n][m]*Cths - FzthB[n][m]*Sths;


if (ind==0) {
```

```
            //Calculate the 3 dipole functionals for the B field

             for (i=0;i<mr;i++) {Bv[i][9] += (BcA[i][n][m]*(FA[0][n][m]*Sths*Cphs[1] + FA[1][n][m]*Cths*Cphs[1] - FA[2][n][m]*Sphs[1]) +
                                              BcB[i][n][m]*(FB[0][n][m]*Sths*Cphs[1] + FB[1][n][m]*Cths*Cphs[1] - FB[2][n][m]*Sphs[1]))*bc[n][m];

                                 Bv[i][10] += (BcA[i][n][m]*(FA[0][n][m]*Sths*Sphs[1] + FA[1][n][m]*Cths*Sphs[1] + FA[2][n][m]*Cphs[1]) +
                                               BcB[i][n][m]*(FB[0][n][m]*Sths*Sphs[1] + FB[1][n][m]*Cths*Sphs[1] + FB[2][n][m]*Cphs[1]))*bc[n][m];

                                 Bv[i][11] += (BcA[i][n][m]*(FA[0][n][m]*Cths - FA[1][n][m]*Sths) +
                                               BcB[i][n][m]*(FB[0][n][m]*Cths - FB[1][n][m]*Sths))*bc[n][m];

                        }

            }


            else{

             //Finally combine odd and even, accumulate for each data point


                        for (i=0;i<mr;i++) {Bv[i][3] += ((BcA[i][n][m]*FxA[0][0][n][m] + BcB[i][n][m]*FxB[0][0][n][m])*dvda[9] +
                                                         (BcA[i][n][m]*FxA[1][0][n][m] + BcB[i][n][m]*FxB[1][0][n][m])*dvda[10] +
                                                         (BcA[i][n][m]*FxA[2][0][n][m] + BcB[i][n][m]*FxB[2][0][n][m])*dvda[11])*bc[n][m];
                                            Bv[i][4] += ((BcA[i][n][m]*FxA[0][1][n][m] + BcB[i][n][m]*FxB[0][1][n][m])*dvda[9] +
                                                         (BcA[i][n][m]*FxA[1][1][n][m] + BcB[i][n][m]*FxB[1][1][n][m])*dvda[10] +
                                                         (BcA[i][n][m]*FxA[2][1][n][m] + BcB[i][n][m]*FxB[2][1][n][m])*dvda[11])*bc[n][m];
                                            Bv[i][5] += ((BcA[i][n][m]*FxA[0][2][n][m] + BcB[i][n][m]*FxB[0][2][n][m])*dvda[9] +
                                                         (BcA[i][n][m]*FxA[1][2][n][m] + BcB[i][n][m]*FxB[1][2][n][m])*dvda[10] +
                                                         (BcA[i][n][m]*FxA[2][2][n][m] + BcB[i][n][m]*FxB[2][2][n][m])*dvda[11])*bc[n][m];
                        }

            }

        }
}



        free(Sphs);

        free(Cphs);

        free(m1);
        free(m2);
        free(m3);

        nmat=end+3;
        mmat=end+3;
        free2darray(Ps,&nmat);
        free2darray(Pdiff1,&nmat);
        free2darray(Pdiff2,&nmat);
        free2darray(Pdiff3,&nmat);

        nmat=end;
        m=3;

        free3darray(FA,&m,&nmat);
        free4darray(FrA,&m,&m,&nmat);
        free4darray(FxA,&m,&m,&nmat);

        free2darray(FxrA,&nmat);
        free2darray(FxthA,&nmat);
        free2darray(FxphA,&nmat);
        free2darray(FyrA,&nmat);
        free2darray(FythA,&nmat);
        free2darray(FyphA,&nmat);
        free2darray(FzrA,&nmat);
        free2darray(FzthA,&nmat);
        free2darray(FzphA,&nmat);

        free3darray(FB,&m,&nmat);
        free4darray(FrB,&m,&m,&nmat);
        free4darray(FxB,&m,&m,&nmat);

        free2darray(FxrB,&nmat);
        free2darray(FxthB,&nmat);
```

```
    free2darray(FxphB,&nmat);
    free2darray(FyrB,&nmat);
    free2darray(FythB,&nmat);
    free2darray(FyphB,&nmat);
    free2darray(FzrB,&nmat);
    free2darray(FzthB,&nmat);
    free2darray(FzphB,&nmat);


    return;
}
```

```
/**** fharmmcg3b
        Magnetic field from Quadrupole volume currents****/


#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "nrutil.h"


float fact(int);

void octapolea          (int,float ***,float ****,float ****,float *****,float *****,float **,int,float,float,float,float *,float *,float *,float *,float *,
                                        float **,float **,float **,int,int);

void octapoleb          (int,float ***,float ****,float ****,float *****,float *****,float **,int,float,float,float,float *,float *,float *,float *,float *,
                                        float **,float **,float **,int,int);

void makeFrr(float *****,float **,int,float,float,float *,float *,float *,float **,float **,float **,int);

/****** Quadrupole version *****/

void fharmmcg3b(int ind,float a[],float **Bv,float ***BcA,float ***BcB,float dvda[],int start,int end,float **bc,int mr)
{
        /****************** Forward MCG field Bv(xyz) on the surface of a sphere due to
                                                                volume current sources from an internal arbitrarily located quadrupole.
                                                                Sphere radius = 1
                                                                Harmonic series version.

                                                                This version computes the functional differentials of the quadrupole ******************/

        //Note that the quadrupole strengths are held in dvda[nc] and the potential
        //is calculated from Qxx,Qxy,Qxz,Qyx ... Qzz held in  dvda[0...8]
        //flagxyz= 1 2 or 3 and indicates whether Bc (the surface constants are for Bx By or Bz
        //Bv is now Bv[i][x,y or z]

        /*** How the differentials are built up.

                        For each term there is an [n][m] and an even (A) and an odd (B) series:

                        Fr, Fth and Fph are d/dr d/dth and d/dph of F {eg A seires FA[0][n][m],FA[1][n][m],FA[2][n][m]}
                        Frr to Fphph are d/dr to d/dph of Fr to Fph {eg A seires FrA[0][0][n][m] thru FrA[2][2][n][m]}
                        Frrr to Fphphph are d/dr d/dth and d/dph of Frr to Fphph {generated by fnc octapole}

                        Fx, Fy and Fz are the cartesian assembly of Fr, Fth and Fph

                        Fxr to Fzph are d/dr to d/dph of Fx to Fz
                        Fxx to Fzz are the cartesian assembly of Fxr to Fzph {eg A series Fx[0][0][n][m] thru Fx[2][2][n][m]}

                        Fxrr to Fzphph are d/dr to d/dph of Fxr to Fzph
                        Fxxr to Fzzph are d/dr to d/dph of Fxx to Fzz
                        Fxxx to Fzzz are the cartesian assembly of Fxxr to Fzzph

                        The final series can be combined with [n][m]  A and B  surface constant terms BcA and BcB  ***/


        int i,j,k,n,m=3,nmat,mmat;
        float r,asph[3],Sths,Cths,*Sphs,*Cphs,*rn1,*rn2,*rn3,**Ps,**LmbdA,**LmbdB,ctcp,stsp,ctsp,stcp,
                **Pdiff1,**Pdiff2,**Pdiff3,***FA,***FB,
                ****FxA,
                ****FxB,

                *****FxxA,
                *****FxxB,

                ****FrA,
                ****FrB,

                *****Frr,

                **FxrA, **FxthA, **FxphA,**FyrA, **FythA, **FyphA, **FzrA, **FzthA,**FzphA,
                **FxrB, **FxthB, **FxphB,**FyrB, **FythB, **FyphB, **FzrB, **FzthB,**FzphB;


        /******* The multipole order is k ************/
```

```
Sphs=malloc((end+1)*sizeof(float));

Cphs=malloc((end+1)*sizeof(float));

rn1=malloc((end+1)*sizeof(float));
rn2=malloc((end+1)*sizeof(float));
rn3=malloc((end+1)*sizeof(float));

nmat=end+3;
mmat=end+3;
Ps=make2darray(&nmat,&mmat);
Pdiff1=make2darray(&nmat,&mmat);

Pdiff2=make2darray(&nmat,&mmat);
Pdiff3=make2darray(&nmat,&mmat);

nmat=end;
mmat=end;
m=3;

FA=make3darray(&m,&nmat,&mmat);

FrA=make4darray(&m,&m,&nmat,&mmat);
FxA=make4darray(&m,&m,&nmat,&mmat);

FxrA=make2darray(&nmat,&mmat);
FxthA=make2darray(&nmat,&mmat);
FxphA=make2darray(&nmat,&mmat);
FyrA=make2darray(&nmat,&mmat);
FythA=make2darray(&nmat,&mmat);
FyphA=make2darray(&nmat,&mmat);
FzrA=make2darray(&nmat,&mmat);
FzthA=make2darray(&nmat,&mmat);
FzphA=make2darray(&nmat,&mmat);

FB=make3darray(&m,&nmat,&mmat);

FrB=make4darray(&m,&m,&nmat,&mmat);
FxB=make4darray(&m,&m,&nmat,&mmat);

FxrB=make2darray(&nmat,&mmat);
FxthB=make2darray(&nmat,&mmat);
FxphB=make2darray(&nmat,&mmat);
FyrB=make2darray(&nmat,&mmat);
FythB=make2darray(&nmat,&mmat);
FyphB=make2darray(&nmat,&mmat);
FzrB=make2darray(&nmat,&mmat);
FzthB=make2darray(&nmat,&mmat);
FzphB=make2darray(&nmat,&mmat);

Frr=make5darray(&m,&m,&m,&nmat,&mmat);
FxxA=make5darray(&m,&m,&m,&nmat,&mmat);
FxxB=make5darray(&m,&m,&m,&nmat,&mmat);

//Ensure Bv is zero
if(ind==0){
        for (i=0;i<mr;i++) {
                for (j=0;j<9;j++) Bv[i][j]= 0;
                }
}
else{
        for (i=0;i<mr;i++) {
                for (j=0;j<3;j++) Bv[i][j]= 0;
                }

}


//convert to spherical
//Note the a values are constant and should be done in main

xyztosph(a,asph);
r=asph[0];

Sths=sin(asph[1]);

Cths=cos(asph[1]);
```

```
        for (i=0;i<=end;i++) {

         Sphs[i]=sin(i*asph[2]);

         Cphs[i]=cos(i*asph[2]);

         rm1[i]=pow(r,(i-1));
         rm2[i]=pow(r,(i-2));
         rm3[i]=pow(r,(i-3));
        }

        nmat=end+3;
        mmat=end+3;
        lgnd(Ps,Cths,nmat);

        stsp=Sths*Sphs[1];
        ctcp=Cths*Cphs[1];
        stcp=Sths*Cphs[1];
        ctsp=Cths*Sphs[1];


//Compute Legendre differentials

/**** New version - allows for Sths = 0 ***/
 for (n=0;n<=(end);n++){
            for (m=0;m<=n;m++){
                    if (m==0) Pdiff1[n][m] = -Ps[n][1];
                    else if (n<=(end+2)) {
                                    Pdiff1[n][m]=((n+m)*(n-m+1)*Ps[n][m-1])/2;
                                    if (m<n) Pdiff1[n][m] -= (Ps[n][m+1])/2;
                            }
                            if (m==0) Pdiff2[n][m] = (Ps[n][2] -n*(n+1)*Ps[n][0])/2;
                            else if (m==1) Pdiff2[n][m] = -((3*n*n+3*n-2)*Ps[n][1] - Ps[n][3])/4;
                            else Pdiff2[n][m] = (((n+m-1)*(n-m+2)*Ps[n][m-2] - Ps[n][m])*(n+m)*(n-m+1) - (n+m+1)*(n-m)*Ps[n][m] - Ps[n][m+2])/4;

                            if (m==0) Pdiff3[n][m] = ((3*n*n+3*n-2)*Ps[n][1] - Ps[n][3])/4;
                            else if (m==1) Pdiff3[n][m] = -(3*n*n+3*n-2)*n*(n+1)*Ps[n][0]/8 + (n+2)*(n-1)*Ps[n][3]/2 -Ps[n][4]/8;
                            else if (m==2) Pdiff3[n][m] = ((3*n*n+3*n+20)*Ps[n][3] - Ps[n][5] - 4*(n+2)*(n-1)*(n+2)*(n-1)*Ps[n][1])/8;
                            else Pdiff3[n][m] = ((n+m-1)*(n-m+2)*(n+m-2)*(n-m+3)*Ps[n][m-3] - (3*n*n+3*n-3*m*m+3*m-2)*Ps[n][m-1])*(n+m)*(n-m+1)/8
                                                                                    + ((3*n*n+3*n-3*m*m-3*m-2)*Ps[n][m+1])/8;


                    }
                   }


/**** Old version ******
            for (n=0;n<=end;n++){
                    printf("\nn= %d ",n);
              for (m=0;m<=n;m++){
                    if (n<=(end+2)) {Pdiff1[n][m]=-(n+1)*Ps[n][m]*Cths/Sths + (n-m+1)*Ps[n+1][m]/Sths;}

                    if ((n>1)&&(n<=(end+1))) Pdiff2[n-1][m]= (-(n-m)*Ps[n][m]+(n-m)*Pdiff1[n][m]*Sths/Cths +
                                                    n*Ps[n-1][m]/Cths - n*Pdiff1[n-1][m]*Sths)*Cths/(Sths*Sths);
                    if ((n>2)&&(n<=(end+2))) Pdiff3[n-2][m]= (n-1-m)*(Pdiff2[n-1][m] - 2*Pdiff1[n-1][m]*Cths/Sths +
                                                    Ps[n-1][m]*(1+2*Cths*Cths/(Sths*Sths)))/Sths +
                                                    (n-1)*(-2*Ps[n-2][m]*Cths/Sths+2*Pdiff1[n-2][m]-Pdiff2[n-2][m]*Sths*Cths)/(Sths*Sths);
                    printf("%8.1f ",Pdiff2[n][m]);

              }
            }

            printf("\n");***********/

/*** Sths must not be zero in this version***/

            for (n=start;n<=end;n++){
              for (m=0;m<=n;m++){

                    FA[0][n][m]= n*rm1[n]*Ps[n][m];
                    FA[1][n][m]=rm1[n]*Pdiff1[n][m];
                    FA[2][n][m]=m*rm1[n]*Ps[n][m]/Sths;

                    FB[0][n][m] = FA[0][n][m]*Sphs[m];
                    FA[0][n][m] *= Cphs[m];

                    FB[1][n][m] = FA[1][n][m]*Sphs[m];
                    FA[1][n][m] *= Cphs[m];
```

```
FB[2][n][m] = FA[2][n][m]*Cphs[m];
FA[2][n][m] *= -Sphs[m];

FrA[0][0][n][m]=n*(n-1)*rn2[n]*Ps[n][m];
FrA[0][1][n][m]=n*rn2[n]*Pdiff1[n][m];
FrA[0][2][n][m]=n*m*rn2[n]*Ps[n][m]/Sths;


FrA[1][0][n][m]=(n-1)*m2[n]*Pdiff1[n][m];
FrA[1][1][n][m]=rn2[n]*Pdiff2[n][m];
FrA[1][2][n][m]=m*rn2[n]*Pdiff1[n][m]/Sths;


FrA[2][0][n][m]=(n-1)*m*rn2[n]*Ps[n][m]/Sths;
FrA[2][1][n][m]=m*rn2[n]*(-Cths*Ps[n][m]/Sths+Pdiff1[n][m])/Sths;
FrA[2][2][n][m]=-m*m*rn2[n]*Ps[n][m]/(Sths*Sths);

FrB[0][0][n][m]= FrA[0][0][n][m]*Sphs[m];
FrB[0][1][n][m]= FrA[0][1][n][m]*Sphs[m];
FrB[0][2][n][m]= FrA[0][2][n][m]*Cphs[m];

FrB[1][0][n][m]= FrA[1][0][n][m]*Sphs[m];
FrB[1][1][n][m]= FrA[1][1][n][m]*Sphs[m];
FrB[1][2][n][m]= FrA[1][2][n][m]*Cphs[m];

FrB[2][0][n][m]= FrA[2][0][n][m]*Cphs[m];
FrB[2][1][n][m]= FrA[2][1][n][m]*Cphs[m];
FrB[2][2][n][m]= FrA[2][2][n][m]*Sphs[m];

FrA[0][0][n][m] *= Cphs[m];
FrA[0][1][n][m] *= Cphs[m];
FrA[0][2][n][m] *= -Sphs[m];

FrA[1][0][n][m] *= Cphs[m];
FrA[1][1][n][m] *= Cphs[m];
FrA[1][2][n][m] *= -Sphs[m];

FrA[2][0][n][m] *= -Sphs[m];
FrA[2][1][n][m] *= -Sphs[m];
FrA[2][2][n][m] *= Cphs[m];

FxrA[n][m] = FrA[0][0][n][m]*Sths*Cphs[1] + FrA[1][0][n][m]*Cths*Cphs[1] - FrA[2][0][n][m]*Sphs[1];
FyrA[n][m] = FrA[0][0][n][m]*Sths*Sphs[1] + FrA[1][0][n][m]*Cths*Sphs[1] + FrA[2][0][n][m]*Cphs[1];
FzrA[n][m] = FrA[0][0][n][m]*Cths - FrA[1][0][n][m]*Sths;

FxthA[n][m] = (FrA[0][1][n][m] - FA[1][n][m]/r)*Sths*Cphs[1] + (FrA[1][1][n][m] + FA[0][n][m]/r)*Cths*Cphs[1] - FrA[2][1][n][m]*Sphs[1];
FythA[n][m] = (FrA[0][1][n][m] - FA[1][n][m]/r)*Sths*Sphs[1] + (FrA[1][1][n][m] + FA[0][n][m]/r)*Cths*Sphs[1] + FrA[2][1][n][m]*Cphs[1];
FzthA[n][m] = (FrA[0][1][n][m] - FA[1][n][m]/r)*Cths - (FrA[1][1][n][m] + FA[0][n][m]/r)*Sths;

FxphA[n][m] = FrA[0][2][n][m]*Sths*Cphs[1] + FrA[1][2][n][m]*Cths*Cphs[1] - FrA[2][2][n][m]*Sphs[1] -
                                    FA[0][n][m]*Sphs[1]/r - FA[1][n][m]*Cths*Sphs[1]/(r*Sths) - FA[2][n][m]*Cphs[1]/(r*Sths);
FyphA[n][m] = FrA[0][2][n][m]*Sths*Sphs[1] + FrA[1][2][n][m]*Cths*Sphs[1] + FrA[2][2][n][m]*Cphs[1] +
                                    FA[0][n][m]*Cphs[1]/r + FA[1][n][m]*Cths*Cphs[1]/(r*Sths) - FA[2][n][m]*Sphs[1]/(r*Sths);
FzphA[n][m] = FrA[0][2][n][m]*Cths - FrA[1][2][n][m]*Sths;


FxrB[n][m] = FrB[0][0][n][m]*Sths*Cphs[1] + FrB[1][0][n][m]*Cths*Cphs[1] - FrB[2][0][n][m]*Sphs[1];
FyrB[n][m] = FrB[0][0][n][m]*Sths*Sphs[1] + FrB[1][0][n][m]*Cths*Sphs[1] + FrB[2][0][n][m]*Cphs[1];
FzrB[n][m] = FrB[0][0][n][m]*Cths - FrB[1][0][n][m]*Sths;

FxthB[n][m] = (FrB[0][1][n][m] - FB[1][n][m]/r)*Sths*Cphs[1] + (FrB[1][1][n][m] + FB[0][n][m]/r)*Cths*Cphs[1] - FrB[2][1][n][m]*Sphs[1];
FythB[n][m] = (FrB[0][1][n][m] - FB[1][n][m]/r)*Sths*Sphs[1] + (FrB[1][1][n][m] + FB[0][n][m]/r)*Cths*Sphs[1] + FrB[2][1][n][m]*Cphs[1];
FzthB[n][m] = (FrB[0][1][n][m] - FB[1][n][m]/r)*Cths - (FrB[1][1][n][m] + FB[0][n][m]/r)*Sths;

FxphB[n][m] = FrB[0][2][n][m]*Sths*Cphs[1] + FrB[1][2][n][m]*Cths*Cphs[1] - FrB[2][2][n][m]*Sphs[1] -
                                    FB[0][n][m]*Sphs[1]/r - FB[1][n][m]*Cths*Sphs[1]/(r*Sths) - FB[2][n][m]*Cphs[1]/(r*Sths);
FyphB[n][m] = FrB[0][2][n][m]*Sths*Sphs[1] + FrB[1][2][n][m]*Cths*Sphs[1] + FrB[2][2][n][m]*Cphs[1] +
                                    FB[0][n][m]*Cphs[1]/r + FB[1][n][m]*Cths*Cphs[1]/(r*Sths) - FB[2][n][m]*Sphs[1]/(r*Sths);;
FzphB[n][m] = FrB[0][2][n][m]*Cths - FrB[1][2][n][m]*Sths;


FxA[0][0][n][m] = FxrA[n][m]*Sths*Cphs[1] + FxthA[n][m]*Cths*Cphs[1] - FxphA[n][m]*Sphs[1];
FxA[1][0][n][m] = FxrA[n][m]*Sths*Sphs[1] + FxthA[n][m]*Cths*Sphs[1] + FxphA[n][m]*Cphs[1];
FxA[2][0][n][m] = FxrA[n][m]*Cths - FxthA[n][m]*Sths;

FxA[0][1][n][m] = FyrA[n][m]*Sths*Cphs[1] + FythA[n][m]*Cths*Cphs[1] - FyphA[n][m]*Sphs[1];
FxA[1][1][n][m] = FyrA[n][m]*Sths*Sphs[1] + FythA[n][m]*Cths*Sphs[1] + FyphA[n][m]*Cphs[1];
```

```
FxA[2][1][n][m] = FyrA[n][m]*Cths - FythA[n][m]*Sths;

FxA[0][2][n][m] = FzrA[n][m]*Sths*Cphs[1] + FzthA[n][m]*Cths*Cphs[1] - FzphA[n][m]*Sphs[1];
FxA[1][2][n][m] = FzrA[n][m]*Sths*Sphs[1] + FzthA[n][m]*Cths*Sphs[1] + FzphA[n][m]*Cphs[1];
FxA[2][2][n][m] = FzrA[n][m]*Cths - FzthA[n][m]*Sths;

FxB[0][0][n][m] = FxrB[n][m]*Sths*Cphs[1] + FxthB[n][m]*Cths*Cphs[1] - FxphB[n][m]*Sphs[1];
FxB[1][0][n][m] = FxrB[n][m]*Sths*Sphs[1] + FxthB[n][m]*Cths*Sphs[1] + FxphB[n][m]*Cphs[1];
FxB[2][0][n][m] = FxrB[n][m]*Cths - FxthB[n][m]*Sths;

FxB[0][1][n][m] = FyrB[n][m]*Sths*Cphs[1] + FythB[n][m]*Cths*Cphs[1] - FyphB[n][m]*Sphs[1];
FxB[1][1][n][m] = FyrB[n][m]*Sths*Sphs[1] + FythB[n][m]*Cths*Sphs[1] + FyphB[n][m]*Cphs[1];
FxB[2][1][n][m] = FyrB[n][m]*Cths - FythB[n][m]*Sths;

FxB[0][2][n][m] = FzrB[n][m]*Sths*Cphs[1] + FzthB[n][m]*Cths*Cphs[1] - FzphB[n][m]*Sphs[1];
FxB[1][2][n][m] = FzrB[n][m]*Sths*Sphs[1] + FzthB[n][m]*Cths*Sphs[1] + FzphB[n][m]*Cphs[1];
FxB[2][2][n][m] = FzrB[n][m]*Cths - FzthB[n][m]*Sths;


if (ind==0) {
//Calculate the 9 quadrupole functionals for the B field

 for (i=0;i<mr;i++) {
        for(j=0; j<3; j++) {
                for(k=0; k<3; k++) {Bv[i][3*j+k] += (BcA[i][n][m]*FxA[k][j][n][m] + BcB[i][n][m]*FxB[k][j][n][m])*bc[n][m];}
        }
}

}


else{

/*** Calculate octapole terms for functional differentials***/

makeFrr(Frr,Ps,m,Sths,Cths,rn1,rn2,rn3,Pdiff1,Pdiff2,Pdiff3,n);
octapolea(nmat,FA,FrA,FxA,Frr,FxxA,Ps,m,r,Sths,Cths,Sphs,Cphs,rn1,rn2,rn3,Pdiff1,Pdiff2,Pdiff3,mr,n);
octapoleb(nmat,FB,FrB,FxB,Frr,FxxB,Ps,m,r,Sths,Cths,Sphs,Cphs,rn1,rn2,rn3,Pdiff1,Pdiff2,Pdiff3,mr,n);

 //Finally combine odd and even, accumulate for each data point

for (i=0;i<mr;i++) {Bv[i][0] +=((BcA[i][n][m]*FxxA[0][0][0][n][m] + BcB[i][n][m]*FxxB[0][0][0][n][m])*dvda[0] +
                                (BcA[i][n][m]*FxxA[1][0][0][n][m] + BcB[i][n][m]*FxxB[1][0][0][n][m])*dvda[1] +
                                (BcA[i][n][m]*FxxA[2][0][0][n][m] + BcB[i][n][m]*FxxB[2][0][0][n][m])*dvda[2] +
                                (BcA[i][n][m]*FxxA[0][1][0][n][m] + BcB[i][n][m]*FxxB[0][1][0][n][m])*dvda[3] +
                                (BcA[i][n][m]*FxxA[1][1][0][n][m] + BcB[i][n][m]*FxxB[1][1][0][n][m])*dvda[4] +
                                (BcA[i][n][m]*FxxA[2][1][0][n][m] + BcB[i][n][m]*FxxB[2][1][0][n][m])*dvda[5] +
                                (BcA[i][n][m]*FxxA[0][2][0][n][m] + BcB[i][n][m]*FxxB[0][2][0][n][m])*dvda[6] +
                                (BcA[i][n][m]*FxxA[1][2][0][n][m] + BcB[i][n][m]*FxxB[1][2][0][n][m])*dvda[7] +
                                (BcA[i][n][m]*FxxA[2][2][0][n][m] + BcB[i][n][m]*FxxB[2][2][0][n][m])*dvda[8])*bc[n][m];
                Bv[i][1] +=((BcA[i][n][m]*FxxA[0][0][1][n][m] + BcB[i][n][m]*FxxB[0][0][1][n][m])*dvda[0] +
                                (BcA[i][n][m]*FxxA[1][0][1][n][m] + BcB[i][n][m]*FxxB[1][0][1][n][m])*dvda[1] +
                                (BcA[i][n][m]*FxxA[2][0][1][n][m] + BcB[i][n][m]*FxxB[2][0][1][n][m])*dvda[2] +
                                (BcA[i][n][m]*FxxA[0][1][1][n][m] + BcB[i][n][m]*FxxB[0][1][1][n][m])*dvda[3] +
                                (BcA[i][n][m]*FxxA[1][1][1][n][m] + BcB[i][n][m]*FxxB[1][1][1][n][m])*dvda[4] +
                                (BcA[i][n][m]*FxxA[2][1][1][n][m] + BcB[i][n][m]*FxxB[2][1][1][n][m])*dvda[5] +
                                (BcA[i][n][m]*FxxA[0][2][1][n][m] + BcB[i][n][m]*FxxB[0][2][1][n][m])*dvda[6] +
                                (BcA[i][n][m]*FxxA[1][2][1][n][m] + BcB[i][n][m]*FxxB[1][2][1][n][m])*dvda[7] +
                                (BcA[i][n][m]*FxxA[2][2][1][n][m] + BcB[i][n][m]*FxxB[2][2][1][n][m])*dvda[8])*bc[n][m];
                Bv[i][2] +=((BcA[i][n][m]*FxxA[0][0][2][n][m] + BcB[i][n][m]*FxxB[0][0][2][n][m])*dvda[0] +
                                (BcA[i][n][m]*FxxA[1][0][2][n][m] + BcB[i][n][m]*FxxB[1][0][2][n][m])*dvda[1] +
                                (BcA[i][n][m]*FxxA[2][0][2][n][m] + BcB[i][n][m]*FxxB[2][0][2][n][m])*dvda[2] +
                                (BcA[i][n][m]*FxxA[0][1][2][n][m] + BcB[i][n][m]*FxxB[0][1][2][n][m])*dvda[3] +
                                (BcA[i][n][m]*FxxA[1][1][2][n][m] + BcB[i][n][m]*FxxB[1][1][2][n][m])*dvda[4] +
                                (BcA[i][n][m]*FxxA[2][1][2][n][m] + BcB[i][n][m]*FxxB[2][1][2][n][m])*dvda[5] +
                                (BcA[i][n][m]*FxxA[0][2][2][n][m] + BcB[i][n][m]*FxxB[0][2][2][n][m])*dvda[6] +
                                (BcA[i][n][m]*FxxA[1][2][2][n][m] + BcB[i][n][m]*FxxB[1][2][2][n][m])*dvda[7] +
                                (BcA[i][n][m]*FxxA[2][2][2][n][m] + BcB[i][n][m]*FxxB[2][2][2][n][m])*dvda[8])*bc[n][m];

        }


}

    }
}
```

```
free(Sphs);

free(Cphs);

free(rn1);
free(rn2);
free(rn3);

nmat=end+3;
mmat=end+3;
free2darray(Ps,&nmat);
free2darray(Pdiff1,&nmat);
free2darray(Pdiff2,&nmat);
free2darray(Pdiff3,&nmat);



nmat=end;
m=3;

free3darray(FA,&m,&nmat);
free4darray(FrA,&m,&m,&nmat);
free4darray(FxA,&m,&m,&nmat);

free2darray(FxrA,&nmat);
free2darray(FxthA,&nmat);
free2darray(FxphA,&nmat);
free2darray(FyrA,&nmat);
free2darray(FythA,&nmat);
free2darray(FyphA,&nmat);
free2darray(FzrA,&nmat);
free2darray(FzthA,&nmat);
free2darray(FzphA,&nmat);

free3darray(FB,&m,&nmat);
free4darray(FrB,&m,&m,&nmat);
free4darray(FxB,&m,&m,&nmat);

free2darray(FxrB,&nmat);
free2darray(FxthB,&nmat);
free2darray(FxphB,&nmat);
free2darray(FyrB,&nmat);
free2darray(FythB,&nmat);
free2darray(FyphB,&nmat);
free2darray(FzrB,&nmat);
free2darray(FzthB,&nmat);
free2darray(FzphB,&nmat);

free5darray(Frr,&m,&m,&m,&nmat);
free5darray(FxxA,&m,&m,&m,&nmat);
free5darray(FxxB,&m,&m,&m,&nmat);


return;
}
```

```
/**** fdipmcg
        Infinite medium magnetic field from a dipole*****/

#include <math.h>
#include<stdio.h>


void fdipmcg(float x[],float a[],float *v)
{
        /****************** Forward MCG inf medium B field vectors Bx,By,Bz the surface of a sphere from
                                                        an internal arbitrarily located dipole.  Sphere radius = 1
 Dipole matrix:         |       Bx              |       By              |       Bz
 Dx     Dy     Dz      |       Fz      -Fy     |       Fx      -Fz      |       Fy
                       |                       |                       |
                       |       v[0]    v[1]    |       v[2]    v[3]    |       v[4]    v[5]

                       x[] is field coords, a[] is source ******************/

        int i,j;
        float cdiff[3],F[3],rho,rho2,rho3,rho5,rho7,
                                gma,rgma,rgma2,rgma3;
        //Define local vars
        cdiff[0]=x[0]-a[0];
        cdiff[1]=x[1]-a[1];
        cdiff[2]=x[2]-a[2];

        //Define globals
        rho=pow((rho2=cdiff[0]*cdiff[0]+cdiff[1]*cdiff[1]+cdiff[2]*cdiff[2]),0.5);
        gma=x[0]*a[0]+x[1]*a[1]+x[2]*a[2];
        rgma=rho+1-gma;


                //Compute lead vector
                for (i=0;i<3;i++) {F[i] = ((2*cdiff[i]/rho2) + x[i] + ((x[i]*gma)-a[i])/rgma)/rho;
                                //remove boundary correction
                                F[i] /= 3;
                        }


                v[0]=F[2];  v[1]=-F[1];  v[2]=F[0];   v[3]=-F[2];  v[4]=F[1];   v[5]=-F[0];

                //printf("%12.8f %12.8f %12.8f\n",cdiff[1],rho2,x[1]);
                //printf("%12.8f %12.8f %12.8f\n",gma,rgma,rho);
                //printf("\n");

        return;
}
```

```
/**** fquad1mcg
        Infinite medium magnetic field from a quadrupole *****/


#include <math.h>


void fquad1mcg(float x[],float a[],float *v)
{
        /****************** Forward MCG inf medium B field vectors Bx,By,Bz the surface of a sphere from
                                                an internal arbitrarily located quadrupole.  Sphere radius = 1

 Quadrupole matrix:             |       Bx              |       By              |       Bz
Qxx     Qyx     Qzx     |-Fzy   -       -       |Fxz+Fzx  Fyz     Fzz     |-Fxy   -Fyy    -Fzy
Qxy     Qyy     Qzy     |-Fxz   -       -Fzz    |  -      -       -       | Fxx   -        Fzx
Qxz     Qyz     Qzz     | Fxy   Fyy     -       |-Fxx     -Fyx    -       |-      -        -
                        |                       |                       |
                        |v[0]   -       -       |v[5]     v[6]    v[7]    |v[10]  v[11]   v[12]
                        | v[1]  -       v[2]    |-        -       -       |v[13]  -       v[14]
                        |v[3]   v[4]    -       |v[8]     v[9]     -      |-      -        -

                        The 8 Quadrupole strengths are held in Q[nc] ******************/

        int i,j;
        float xdiff,ydiff,zdiff,xidiff,xjdiff,F[3][3],rho,rho2,rho3,rho5,rho7,
                        gma,rgma,rgma2,rgma3;
        //Define local vars
        xdiff=x[0]-a[0];
        ydiff=x[1]-a[1];
        zdiff=x[2]-a[2];

        //Define globals
        rho=pow((rho2=xdiff*xdiff+ydiff*ydiff+zdiff*zdiff),0.5);
        rho3=pow(rho,3);
        rho5=pow(rho,5);
        rho7=pow(rho,7);
        gma=x[0]*a[0]+x[1]*a[1]+x[2]*a[2];
        rgma=rho+1-gma;
        rgma2=pow(rgma,2);
        rgma3=pow(rgma,3);


                //Compute surface magnetic field for infinite medium
                for (i=0;i<3;i++) {
                        for(j=0;j<3;j++) {
                                xidiff=x[i]-a[i]; xjdiff=x[j]-a[j];
                                F[i][j] = (i==j)*(-2)/rho3
                                        + 6*xidiff*xjdiff/rho5
                                        + x[i]*xjdiff/rho3
                                        + (x[i]*x[j] - (i==j))/(rho*rgma)
                                        + ((x[i]*gma)-a[i])*(xjdiff*(rho+rgma) + rho2*x[j])/(rho3*rgma*rgma);

                                //remove boundary correction
                                F[i][j] *= (2.0/5);
                        }
                }

                v[0]=-F[2][1];          v[1]=-F[0][2];          v[2]=-F[2][2];          v[3]=F[0][1];           v[4]=F[1][1];
                v[5]=F[0][2]+F[2][0];   v[6]=F[1][2];           v[7]=F[2][2];           v[8]=-F[0][0];          v[9]=-F[1][0];
                v[10]=-F[0][1];         v[11]=-F[1][1];         v[12]=-F[2][1];         v[13]=F[0][0];          v[14]=F[2][0];

                        //For testing
                        /*Bd[0] = -Q[1]*F[0][2]+Q[2]*F[0][1]+Q[5]*(F[1][1]-F[2][2])-Q[0]*F[2][1]+Q[4]*(-F[2][1]-F[1][2]);
                        Bd[1] = Q[1]*F[1][2]+Q[2]*(F[2][2]-F[0][0])-Q[5]*F[1][0]+Q[0]*(F[0][2]+F[2][0])+Q[4]*F[2][0];
                        Bd[2] = Q[1]*(F[0][0]-F[1][1])-Q[2]*F[2][1]+Q[5]*F[2][0]-Q[0]*F[0][1]+Q[4]*F[1][0];*/


        return;
}
```

```
/**** fdip0mcg
        First differential of magnetic field functionals for a dipole ****/


#include <math.h>


void fdip0mcg(int flagxyz,float x[],float a[],float *Q,float *d)
{
        /****************** Forward MCG inf medium first differential of B field vectors Bx,By,Bz the surface
                           of a sphere from an internal arbitrarily located dipole.  Sphere radius = 1
  Dipole matrix:                   | Bx                |By                |Bz
Dx        Dy        Dz             | Fzx      -Fyx     |Fxx      -Fzx     |Fyx      -Fxx
                                   | Fzy      -Fyy     |Fxy      -Fzy     |Fyy      -Fxy
                                   | Fzz      -Fyz     |Fxz      -Fzz     |Fyz      -Fxz
                                   |                   |                  |
                                   | F[2][0]  F[1][0]  | F[0][0]  F[2][0]  |F[1][0]  F[0][0]
                                   | F[2][1]  F[1][1]  | F[0][1]  F[2][1]  |F[1][1]  F[0][1]
                                   | F[2][2]  F[1][2]  | F[0][2]  F[2][2]  |F[1][2]  F[0][2]

                                   x[] is field coords, a[] is source ******************/


        int         i,j;
        float       xdiff,ydiff,zdiff,xidiff,xjdiff,F[3][3],rho,rho2,rho3,rho5,gma,rgma;
        //Define local vars
        xdiff=x[0]-a[0];
        ydiff=x[1]-a[1];
        zdiff=x[2]-a[2];

        //Define globals
        rho=pow((rho2=xdiff*xdiff+ydiff*ydiff+zdiff*zdiff),0.5);
        rho3=pow(rho,3);
        rho5=pow(rho,5);

        gma=x[0]*a[0]+x[1]*a[1]+x[2]*a[2];
        rgma=rho+1-gma;


        //Compute surface magnetic field 1st differentials for infinite medium
        //Bx
        if (flagxyz==0) {
                for(i=2;i>0;i--) {
                        for(j=0;j<3;j++) {
                                xidiff=x[i]-a[i]; xjdiff=x[j]-a[j];
                                F[i][j] = (i==j)*(-2)/rho3
                                        + 6*xidiff*xjdiff/rho5
                                        + x[i]*xjdiff/rho3
                                        + (x[i]*x[j] - (i==j))/(rho*rgma)
                                        + ((x[i]*gma)-a[i])*(xjdiff*(rho+rgma) + rho2*x[j])/(rho3*rgma*rgma);

                                //remove boundary correction
                                F[i][j] /= 3;
                        }
                }
                Q[0]=d[1]*F[2][0]-d[2]*F[1][0]; Q[1]=d[1]*F[2][1]-d[2]*F[1][1]; Q[2]=d[1]*F[2][2]-d[2]*F[1][2];
        }

        //By
        if (flagxyz==1) {
                for(i=0;i<3;i+=2) {
                        for(j=0;j<3;j++) {
                                xidiff=x[i]-a[i]; xjdiff=x[j]-a[j];
                                F[i][j] = (i==j)*(-2)/rho3
                                        + 6*xidiff*xjdiff/rho5
                                        + x[i]*xjdiff/rho3
                                        + (x[i]*x[j] - (i==j))/(rho*rgma)
                                        + ((x[i]*gma)-a[i])*(xjdiff*(rho+rgma) + rho2*x[j])/(rho3*rgma*rgma);

                                //remove boundary correction
                                F[i][j] /= 3;
                        }
                }
                Q[0]=d[2]*F[0][0]-d[0]*F[2][0]; Q[1]=d[2]*F[0][1]-d[0]*F[2][1]; Q[2]=d[2]*F[0][2]-d[0]*F[2][2];
        }
        //Bz
        if (flagxyz==2) {
```

```
        for(i=1;i>=0;i--) {
                for(j=0;j<3;j++) {
                        xidiff=x[i]-a[i]; xjdiff=x[j]-a[j];
                        F[i][j] = (i==j)*(-2)/rho3
                                + 6*xidiff*xjdiff/rho5
                                + x[i]*xjdiff/rho3
                                + (x[i]*x[j] - (i==j))/(rho*rgma)
                                + ((x[i]*gma)-a[i])*(xjdiff*(rho+rgma) + rho2*x[j])/(rho3*rgma*rgma);

                        //remove boundary correction
                        F[i][j] /= 3;
                }
        }
        Q[0]=d[0]*F[1][0]-d[1]*F[0][0]; Q[1]=d[0]*F[1][1]-d[1]*F[0][1]; Q[2]=d[0]*F[1][2]-d[1]*F[0][2];
}

return;
}
```

```
#include <math.h>
float Fxxx (float,float,float,float,float,float);
float Fxxy (float,float,float,float,float,float,float,float);
float Fxyz (float,float,float,float,float,float,float,float,float,float);
float Fxyx (float,float,float,float,float,float,float,float);
float Fxyy (float,float,float,float,float,float,float);

void fquad0mcg(int flag,float x[],float a[],float *dvda,float *Q)
{
        /****************** Calculates the partial differentials for the Bx, By and Bz vectors
                for the inverse MCG inf medium non-linear functionals.
                Note that in all cases the Qyy term is linearly dependent in the
                variable projection method and is not computed.  Also Qzz as a
                member of the traceless tensor is not part of the projection.
                Instead,  Qxx assumes different values for each of the separate
                Bx, By and Bz projections.  The true Qxx, Qyy and Qzz values
                can be deduced from any pair of these projections.

 Quadrupole matrix:        | Bx                          |By                          |Bz
Qxx     Qyx     Qzx      |-Fzy     -       -          |Fxz+Fzx   Fyz     Fzz      |-Fxy     -Fyy    -Fzy
Qxy     Qyy     Qzy      |-Fxz     dep     -Fzz       |  -       dep     -        | Fxx     dep     Fzx
Qxz     Qyz     Qzz      | Fxy     Fyy     -          |-Fxx      -Fyx    -        | -       -       -
                         |                             |                            |
                         | v[0]    -       -          |v[5]      v[6]    v[7]     |v[10]    v[11]   v[12]
                         | v[1]    -       v[2]       | -        -       -        |v[13]    -       v[14]
                         | v[3]    v[4]    -          |v[8]      v[9]    -        | -       -       -

                        Quadrupole strengths are held in Q[nc] ******************/

        int i,j,k,pos=0,idx=0;
        float xdiff[3],xsq[3],xgma[3],delx[3],xrho[3],xidiff,xjdiff,old,func[6];
        extern float rho,rho2,rho3,rho4,rho5,rho7,gma,rgma,rgma2,rgma3;
        //Define local and global variables
        gma=0;
        rho2=0;
        for (i=0;i<3;i++) {
                gma += x[i]*a[i];
                xdiff[i]=x[i]-a[i];
                rho2 += xdiff[i]*xdiff[i];
                xsq[i]=x[i]*x[i]-1;
        }

        rho=pow(rho2,0.5);
        rho4=pow(rho2,2);
        rho3=pow(rho,3);
        rho5=pow(rho,5);
        rho7=pow(rho,7);
        rgma=rho+1-gma;
        rgma2=pow(rgma,2);
        rgma3=pow(rgma,3);

        for (i=0;i<3;i++) {
                xgma[i]=x[i]*gma-a[i];
                delx[i]=(xdiff[i]/rho)+x[i];
                xrho[i]=x[i]*rho+xdiff[i];
        }

                //Compute the partial differential tensor
                //each dvda[k] for x(j)=(x,y,z) and for x(i)=(x,y,z) is
                //the summation of d{d(Fx(i))/dx(j)}/dx(k)  for x(k)=(x,y,z)
                //resulting in 3 terms

                /****  Bx partial differentials ( F_y and F_z) ****/
                if (flag==0){
                   for (k=0;k<3;k++) {
                        dvda[k]=0;
                        pos=0;
                        for (j=1;j<3;j++) {
                                for (i=0;i<3;i++) {
                                   if ((i==j)&&(i==k))          func[pos] = Fxxx(xdiff[i],x[i],xrho[i],xsq[i],delx[i],xgma[i]);
                                   else if((i==j)&&(i!=k))      func[pos] = Fxxy(xdiff[i],xdiff[k],x[i],xsq[i],delx[k],delx[i],x[k],xgma[i]);
                                   else if ((i!=j)&&(j!=k)&&(i!=k)) func[pos] =Fxyz(xdiff[i],xdiff[j],xdiff[k],x[i],x[j],x[k],delx[k],delx[j],xgma[i]);
                                   else if ((i!=j)&&(i==k))     func[pos] = Fxyx(xdiff[j],xdiff[i],x[i],delx[i],x[j],xsq[i],delx[j],xgma[i]);
                                   else if  ((i!=j)&&(j==k))    func[pos] = Fxyy(xdiff[i],xdiff[j],x[i],xrho[i],delx[j],x[j],xgma[i]);

                                   //remove boundary correction
                                   func[pos] *= (2.0/5);
```

```
                                        if (pos==0) dvda[k] += Q[6]*func[pos];
                                        else if (pos==1) dvda[k] += Q[7]*func[pos];
                                        else if (pos==2) dvda[k] += -Q[0]*func[pos];
                                        else if (pos==3) dvda[k] += -Q[3]*func[pos];
                                        else if (pos==5) dvda[k] += -Q[5]*func[pos];
                                        else ;

                                        pos++;
                                        }
                                }
                        }
                }

                /**** By partial differentials ( F_x and F_z) ****/
                else if (flag==1) {
                    for (k=0;k<3;k++) {
                            dvda[k]=0;
                            pos=0;
                            for (j=0;j<3;j+=2) {
                                    for (i=0;i<3;i++) {
                                        if ((i==j)&&(i==k))             func[pos] = Fxxx(xdiff[i],x[i],xrho[i],xsq[i],delx[i],xgma[i]);
                                        else if  ((i==j)&&(i!=k))       func[pos] = Fxxy(xdiff[i],xdiff[k],x[i],xsq[i],delx[k],delx[i],x[k],xgma[i]);
                                        else if ((i!=j)&&(j!=k)&&(i!=k)) func[pos] = Fxyz(xdiff[i],xdiff[j],xdiff[k],x[i],x[j],x[k],delx[k],delx[j],xgma[i]);
                                        else if ((i!=j)&&(i==k))        func[pos] = Fxyx(xdiff[j],xdiff[i],x[i],delx[i],x[j],xsq[i],delx[j],xgma[i]);
                                        else if  ((i!=j)&&(j==k))       func[pos] = Fxyy(xdiff[i],xdiff[j],x[i],xrho[i],delx[j],x[j],xgma[i]);

                                        //remove boundary correction
                                        func[pos] *= (2.0/5);

                                        if (pos==0) dvda[k] += -Q[6]*func[pos];
                                        else if (pos==1) dvda[k] += -Q[7]*func[pos];
                                        else if (pos==2) dvda[k] += Q[0]*func[pos];
                                        else if (pos==3) dvda[k] += Q[0]*func[pos];
                                        else if (pos==4) dvda[k] += Q[1]*func[pos];
                                        else     dvda[k] += Q[2]*func[pos];

                                        pos++;
                                        }
                                    }
                            }
                }

                /**** Bz partial differentials ( F_x and F_y) ****/
                else {
                    for (k=0;k<3;k++) {
                            dvda[k]=0;
                            pos=0;
                            for (j=0;j<2;j++) {
                                    for (i=0;i<3;i++) {
                                        if ((i==j)&&(i==k))             func[pos] = Fxxx(xdiff[i],x[i],xrho[i],xsq[i],delx[i],xgma[i]);
                                        else if ((i==j)&&(i!=k))        func[pos] = Fxxy(xdiff[i],xdiff[k],x[i],xsq[i],delx[k],delx[i],x[k],xgma[i]);
                                        else if ((i!=j)&&(j!=k)&&(i!=k)) func[pos] = Fxyz(xdiff[i],xdiff[j],xdiff[k],x[i],x[j],x[k],delx[k],delx[j],xgma[i]);
                                        else if ((i!=j)&&(i==k))        func[pos] = Fxyx(xdiff[j],xdiff[i],x[i],delx[i],x[j],xsq[i],delx[j],xgma[i]);
                                        else if  ((i!=j)&&(j==k))       func[pos] = Fxyy(xdiff[i],xdiff[j],x[i],xrho[i],delx[j],x[j],xgma[i]);

                                        //remove boundary correction
                                        func[pos] *= (2.0/5);

                                        if (pos==0) dvda[k] += Q[3]*func[pos];
                                        else if (pos==2) dvda[k] += Q[5]*func[pos];
                                        else if (pos==3) dvda[k] += -Q[0]*func[pos];
                                        else if (pos==4) dvda[k] += -Q[1]*func[pos];
                                        else if (pos==5) dvda[k] += -Q[2]*func[pos];
                                        else ;

                                        pos++;
                                        }
                                    }
                            }
                }


        return;
}
```

/** Fxxx .... to .... Fxyy ****/**

```
float Fxxx (float xdiff,float xe,float xrho,float xsq,float delx,float xgma)
{
        return (-18*xdiff/rho5 + 30*xdiff*xdiff*xdiff/rho7 + 3*xe*xdiff*xdiff/rho5
                - xrho/(rgma*rho3) + 2*xsq*delx/(rgma2*rho) + 2*xsq*xdiff/(rgma*rho3)
                + 2*xgma*delx*delx/(rgma3*rho) + 2*xgma*xdiff*delx/(rgma2*rho3)
                + xgma*(pow((xdiff/rho),2)-1)/(rgma2*rho2) + 3*xgma*xdiff*xdiff/(rgma*rho5));
}

float Fxxy (float xdiff,float ydiff,float xe,float xsq,float dely,float delx,float ye,float xgma)
{
        return (-6*ydiff/rho5 + 30*xdiff*xdiff*ydiff/rho7 + 3*xe*xdiff*ydiff/rho5
                + xsq*dely/(rgma2*rho) + (xsq*ydiff+xe*xdiff*ye)/(rgma*rho3) + xe*delx*ye/(rgma2*rho)
                + 2*xgma*delx*dely/(rgma3*rho) + xgma*(ydiff*delx+xdiff*dely)/(rgma2*rho3)
                + xgma*xdiff*ydiff/(rgma2*rho4) + 3*xgma*xdiff*ydiff/(rgma*rho5));
}

float Fxyz (float xdiff,float ydiff,float zdiff,float xe,float ye,float ze,float delz,float dely,float xgma)
{
        return (30*xdiff*ydiff*zdiff/rho7 + 6*xe*ydiff*zdiff/rho5
                + xe*ye*delz/(rgma2*rho) + xe*(ye*zdiff+ze*ydiff)/(rgma*rho3) + xe*dely*ze/(rgma2*rho)
                + 2*xgma*dely*delz/(rgma3*rho) + xgma*(ydiff*delz+zdiff*dely)/(rgma2*rho3)
                + xgma*ydiff*zdiff/(rgma2*rho4) + 3*xgma*ydiff*zdiff/(rgma*rho5));
}

float Fxyx (float ydiff,float xdiff,float xe,float delx,float ye,float xsq,float dely,float xgma)
{
        return (-6*ydiff/rho5 + 30*xdiff*ydiff*xdiff/rho7 + 3*xe*xdiff*ydiff/rho5
                + xe*delx*ye/(rgma2*rho) + (xsq*ydiff + xe*xdiff*ye)/(rgma*rho3) +xsq*dely/(rgma2*rho)
                + 2*xgma*delx*dely/(rgma3*rho) + xgma*(ydiff*delx+xdiff*dely)/(rgma2*rho3)
                + xgma*xdiff*ydiff/(rgma2*rho4) + 3*xgma*xdiff*ydiff/(rgma*rho5));
}

float Fxyy (float xdiff,float ydiff,float xe,float xrho,float dely,float ye,float xgma)
{
        return (-6*xdiff/rho5 + 30*xdiff*ydiff*ydiff/rho7 + 3*xe*ydiff*ydiff/rho5 - xrho/(rgma*rho3)
                + 2*xe*dely*ye/(rgma2*rho) + 2*xe*ye*ydiff/(rgma*rho3)
                + 2*xgma*dely*dely/(rgma3*rho) + 2*xgma*ydiff*dely/(rgma2*rho3)
                + xgma*(pow((ydiff/rho),2)-1)/(rgma2*rho2) + 3*xgma*ydiff*ydiff/(rgma*rho5));
}
```

The Utility Functions

/******* nrutil.h**
originally from Numerical Recipes [Vetterling et al] but modified and expanded*******/

```
#include <stdio.h>
#include <stddef.h>
#include <stdlib.h>
#include <math.h>
#define NR_END 1
#define FREE_ARG char*

float fact(int n);

void nrerror(char error_text[])
{
	fprintf(stderr,"Numerical Recipes run-time error ....\n");
	fprintf(stderr,"%s\n",error_text);
	fprintf(stderr,"... now exiting to system ...\n");
	exit(1);
}


void free_ivector(int *v, long nl, long nh)
{
	free((FREE_ARG) (v+nl-NR_END));
}


int *ivector(long nl, long nh)
{
	int *v;

	v=(int *)malloc((size_t) ((nh-nl+1+NR_END)*sizeof(int)));
	if (!v) nrerror("allocation failure in ivector()");
	return v-nl+NR_END;
}

float **make2darray(int *n,int *m)
{
	int i,j;
	float **ma;
	ma=(float **)malloc((1+*n)*sizeof(float*));/*pointers to each of n rows*/

	/*Memory test*/
	if(!ma) {
		printf("memory request failed\n");
		return;
	}

	/*Allocate m column spaces to each row pointer*/
	for (j=0;j<=*n;i++) {
		ma[i]=(float *)malloc((1+*m)*sizeof(float));

		/*Memory test*/
			if(!ma[i]) {
				printf("memory request failed\n");
				return;
			}
	}
	for(i=0;i<=*n;i++) {
		for(j=0;j<=*m;j++) {
			ma[i][j]=0;
		}
	}
	return ma;
}


float ***make3darray(int *p,int *n,int *m)
{
	int i,j,k;
	float ***ma;
	ma=(float ***)malloc((1+*p)*sizeof(float*));/*pointers to each of p matrices*/
```

```
	/*Memory test*/
	if(!ma) {
		printf("memory request failed\n");
		return;
	}

	/*Allocate n row pointers to each p matrix*/
	for (i=0;i<=*p;i++) {
			ma[i]=(float **)malloc((1+*n)*sizeof(float));

			/*Memory test*/
				if(!ma[i]) {
					printf("memory request failed\n");
				return;
				}
	}


	for(i=0;i<=*p;i++) {
		for(j=0;j<=*n;j++) {
			ma[i][j]=(float *)malloc((1+*m)*sizeof(float));

			/*Memory test*/
				if(!ma[i][j]) {
					printf("memory request failed\n");
				return;
				}

		}
	}

	//Initialise to zero
	for(i=0;i<=*p;i++) {
		for(j=0;j<=*n;j++) {
			for(k=0;k<=*m;k++) {
				ma[i][j][k]=0;
			}
		}
	}


	return ma;
}

float ****make4darray(int *q,int *p,int *n,int *m)
{
	int h,i,j,k;
	float ****ma;
	ma=(float ****)malloc((1+*q)*sizeof(float*));/*pointers to each of q matrices*/

	/*Memory test*/
	if(!ma) {
		printf("memory request failed\n");
		return;
	}

	/*Allocate p table pointers to each p matrix*/
	for (h=0;h<=*q;h++) {
			ma[h]=(float ***)malloc((1+*p)*sizeof(float));

			/*Memory test*/
				if(!ma[h]) {
					printf("memory request failed\n");
				return;
				}
	}


	for(h=0;h<=*q;h++) {
		for(i=0;i<=*p;i++) {
			ma[h][i]=(float **)malloc((1+*n)*sizeof(float));

			/*Memory test*/
				if(!ma[h][i]) {
					printf("memory request failed\n");
				return;
				}
		}
```

```
	}

	for(h=0;h<=*q;h++) {
		for(i=0;i<=*p;i++) {
			for (j=0;j<=*n;j++) {
				ma[h][i][j]=(float *)malloc((1+*m)*sizeof(float));

				/*Memory test*/
					if(!ma[h][i][j]) {
						printf("memory request failed\n");
					return;
					}
				}
		}
	}

	//Initialise to zero
	for(h=0;h<=*q;h++) {
		for(i=0;i<=*p;i++) {
			for(j=0;j<=*n;j++) {
				for(k=0;k<=*m;k++) {
					ma[h][i][j][k]=0;
				}
			}
		}
	}

	return ma;
}

float *****make5darray(int *r,int *q,int *p,int *n,int *m)
{
	int g,h,i,j,k,l;
	float *****ma;
	ma=(float *****)malloc((1+*r)*sizeof(float*));/*pointers to each of q matrices*/

	/*Memory test*/
	if(!ma) {
		printf("memory request failed\n");
		return;
	}

	/*Allocate p table pointers to each p matrix*/
	for (g=0;g<=*r;g++) {
			ma[g]=(float ****)malloc((1+*q)*sizeof(float));

			/*Memory test*/
				if(!ma[g]) {
					printf("memory request failed\n");
				return;
				}
	}


	for(g=0;g<=*r;g++) {
		for(h=0;h<=*q;h++) {
			ma[g][h]=(float ***)malloc((1+*p)*sizeof(float));

				/*Memory test*/
					if(!ma[g][h]) {
					printf("memory request failed\n");
					return;
					}
				}
		}

	for(g=0;g<=*r;g++) {
		for(h=0;h<=*q;h++) {
			for (i=0;i<=*p;i++) {
				ma[g][h][i]=(float **)malloc((1+*n)*sizeof(float));

				/*Memory test*/
					if(!ma[g][h][i]) {
					printf("memory request failed\n");
					return;
					}
				}
		}
```

```
		}

		for(g=0;g<=*r;g++) {
			for(h=0;h<=*q;h++) {
				for (i=0;i<=*p;i++) {
					for (j=0;j<=*n;j++) {
						ma[g][h][i][j]=(float *)malloc((1+*m)*sizeof(float));

						/*Memory test*/
						   if(!ma[g][h][i][j]) {
							printf("memory request failed\n");
							return;
						}
					}
				}
			}
		}

		//Initialise to zero
		for(h=0;h<=*r;h++) {
			for(i=0;i<=*q;i++) {
				for(j=0;j<=*p;j++) {
					for(k=0;k<=*n;k++) {
						for(l=0;l<=*m;l++) {
						   ma[h][i][j][k][l]=0;
						}
					}
				}
			}
		}
		return ma;
}


void free2darray(float **mat,int *rows)
{
		int i;
		for (i=0;i<=*rows;i++) {
			free(mat[i]);
			}
		free(mat);

		return;
}

void free3darray(float ***mat,int *tab,int *rows)
{
		int i,j;
		for (i=0;i<=*tab;i++) {
			for (j=0;j<=*rows;j++) {
				free(mat[i][j]);
			}
		}

		for (i=0;i<=*tab;i++) {
				free(mat[i]);
		}

		free(mat);

		return;
}

void free4darray(float ****mat,int *tbtb,int *tab,int *rows)
{
		int i,j,k;
		for (i=0;i<=*tbtb;i++) {
			for (j=0;j<=*tab;j++) {
				for (k=0;k<=*rows;k++) {
					free(mat[i][j][k]);
				}
			}
		}

		for (i=0;i<=*tbtb;i++) {
			for (j=0;j<=*tab;j++) {
				free(mat[i][j]);
			}
```

```
	}

	for (i=0;i<=*tbtb;i++) {
			free(mat[i]);
	}

	free(mat);

	return;
}

void free5darray(float *****mat,int *tb5,int *tbtb,int *tab,int *rows)
{
	int i,j,k,l;
	for (i=0;i<=*tb5;i++) {
		for (j=0;j<=*tbtb;j++) {
			for (k=0;k<=*tab;k++) {
				for (l=0;l<=*rows;l++) {
					free(mat[i][j][k][l]);
				}
			}
		}
	}

	for (i=0;i<=*tb5;i++) {
		for (j=0;j<=*tbtb;j++) {
			for (k=0;k<=*tab;k++) {
				free(mat[i][j][k]);
			}
		}
	}

	for (i=0;i<=*tb5;i++) {
		for (j=0;j<=*tbtb;j++) {
				free(mat[i][j]);
		}
	}

	for (i=0;i<=*tb5;i++) {
			free(mat[i]);
	}

	free(mat);

	return;
}


float *vector(long nl,long nh)
//allocate a float vector with range v[nl...nh]
{
	float *v;

	v=(float *)malloc((size_t) ((nh-nl+1+NR_END)*sizeof(float)));
	if (!v) nrerror("allocatopn failure in vector()");
	return v-nl+NR_END;
}

void free_vector(float *v,long nl,long nh)
{
	free((FREE_ARG) (v+nl-NR_END));
}

float **matrix(long nrl,long nrh,long ncl,long nch)
//allocate a float matrix with range m[nrl..nrh][ncl..nch]
{
	long i,nrow=nrh-nrl+1,ncol=nch-ncl+1;
	float **m;

	//allocate pointers to rows
	m=(float **)malloc((size_t) ((nrow+NR_END)*sizeof(float*)));
	if (!m) nrerror("allocation failure 1 in matrix()");
	m +=NR_END;
	m-=nrl;

	//allocate rows and set pointers to them
	m[nrl]=(float *)malloc((size_t) ((nrow*ncol+NR_END)*sizeof(float)));
	if (!m[nrl]) nrerror("allocation failure 2 in matrix()");
```

```
	m[nrl] +=NR_END;
	m[nrl] -= ncl;

	for(i=nrl+1;i<=nrh;i++) m[i]=m[i-1]+ncol;

	return m;
}

void free_matrix(float **m,long nrl,long nrh,long ncl,long nch)
{
	free((FREE_ARG) (m[nrl]+ncl-NR_END));
	free((FREE_ARG) (m+nrl-NR_END));
}

void xyztosph(float *xyz,float *sph)
{
    //calculate r
    sph[0]=pow((xyz[0]*xyz[0]+xyz[1]*xyz[1]+xyz[2]*xyz[2]),0.5);

    //calculate theta
    sph[1]=acos(xyz[2]/sph[0]);

    //calculate phi
    if (xyz[1]<0) {
	sph[2]= 2*3.141592654-acos(xyz[0]/(sph[0]*sin(sph[1])));
    }
    else if (fabs(xyz[2])==1) sph[2]=0;//3.141592654/2;
    else  sph[2]=acos(xyz[0]/(sph[0]*sin(sph[1])));

    return;
}

void sphtoxyz(float *sph,float *xyz)
{
    //calculate x
    xyz[0]=sph[0]*cos(sph[1]);

    //calculate y
    xyz[1]=sph[0]*sin(sph[1])*sin(sph[2]);

    //calculate z
    xyz[2]=sph[0]*sin(sph[1])*cos(sph[2]);

    return;

}

void rotyz(float *xyz,float deg)
{
	float y,z,rad;

	rad=deg*2*3.141592654/360;
	y=xyz[1]*cos(rad)-xyz[2]*sin(rad);
	z=-xyz[1]*sin(rad)-xyz[2]*cos(rad);

	xyz[1]=y; xyz[2]=z;

	return;
}


void lgnd(float **P,float x,int nx)
{
	//Legendre and Associated Legendre functions
	float sx,ct=1.0;
  int n,m;

	sx=pow((1-x*x),0.5);

//	 ct=(pow(-1,m))*(fact(n-m))/fact(n+m);



	P[0][0]=1;
	P[1][0]=x;
	P[1][1]=sx;

	//Legendre functions
```

```
        for (n=1;n<=(nx-1);n++) P[n+1][0]=((2*n+1)*x*P[n][0]-n*P[n-1][0])/(n+1);

        //Associated Legendre functions
        for (n=1;n<=(nx-1);n++) {
                for (m=0;m<=n;m++) {
                        P[n+1][m+1]=(2*n+1)*sx*P[n][m];
                        if ((m+1)<=(n-1)) P[n+1][m+1] += P[n-1][m+1];
                }
        }

        return;
}

float fact(int n)
{
 int i;
 float q=1.0;

 if (n<=0) return q;
 for (i=1;i<=n;i++) {
        q *= i;
 }

 return q;
}
```

```
/**** inputdata
        various input functions ****/


#include <stdio.h>
#include <stdlib.h>

void ecgdata(FILE *fp,float **x,float *v,int *mr)
{
/********** changed 20/2/98 so that all arrays start from 0 **********/
        //For reading in x,y and z surface cartesian coordinates
        //and the surface potential at that point

        register int i;


        for(i=0;i<*mr;++i) {
                fscanf(fp,"%f %f %f %f\n",x[i],x[i]+1,x[i]+2,&v[i]);
                //printf("%14.9f %f %14.9f %14.9f %14.9f\n",v[i],100*x[i][0],100*x[i][1],100*x[i][2]);
        }


//      printf("array is %d long\n",*mr);

        return;
}

void mcgdata(FILE *fp,float **x,float **B,int *mr)
{

        //For reading in x,y and z surface cartesian coordinates
        //and the surface magnetic field at that point

        register int i;


        for(i=0;i<*mr;++i) {
                fscanf(fp,"%f %f %f %f %f %f\n",x[i],x[i]+1,x[i]+2,B[i],B[i]+1,B[i]+2);
                //printf("%14.9f %14.9f %14.9f %14.9f %14.9f %14.9f\n",B[i][0],B[i][1],B[i][2],x[i][0],x[i][1],x[i][2]);
        }


//      printf("array is %d long\n",*mr);

        return;
}

void mcgdata2(FILE *fp,float **x,float *B,int *mr)
{

        //For reading in x,y and z surface cartesian coordinates
        //and the surface magnetic field at that point for a single plane (Bx or By or Bz)

        register int i;


        for(i=0;i<*mr;++i) {
                fscanf(fp,"%f %f %f %f\n",x[i],x[i]+1,x[i]+2,&B[i]);
                //printf("%f %f %f %f %f %f\n",B[i],x[i][0],x[i][1],x[i][2]);
        }


//      printf("array is %d long\n",*mr);

        return;
}

void coordata(FILE *fp,float **x,int *mr)
{
/********** changed 20/2/98 so that all arrays start from 0 **********/

        //For reading in x,y and z surface cartesian coordinates
        //generated by Dome

        char str1[80],str2[80],str3[80];
        register int i;
```

```
	for(i=0;i<*mr;++i) {
		fscanf(fp,"%s %s %s",str1,str2,str3);
		x[i][0]=atof(str1);
		x[i][1]=atof(str2);
		x[i][2]=atof(str3);
	}

	return;
}


int filesize(char *fname)
{
	//Interrogate file for file size
	FILE *fp;
	char str1[80],str2[80],str3[80];
	register int ndat=0;

	if((fp=fopen(fname,"r")) == NULL) {
					printf("Cannot open file %s\n",fname);
					return;
					}

	//NB This seems to read in the last record twice
	//but ndat needs to be 1 greater than the matrix
	//because the array referencing system is FORTRAN based!
	while (!feof(fp)) {
		fscanf(fp,"%s %s %s",str1,str2,str3);
		ndat++;
	}
	rewind(fp);


	return ndat;
}
```

```
/**** householder
        Householder decomposition function ****/


#include <stdio.h>
#include <math.h>
#define SWAP(a,b)       {float temp=a;a=b;b=temp;}
#define SIGNOF(a)       a/fabs(a)

void househ(float **A,int *mr,int *nc,int *rnk,int *s,float **v,float *delta,int *sbreak,float *rbeta)
{
        /***** The Householder process is controlled by a permutation vector,s,
        which contains the sequence in which the column vectors of A are to be
        treated.  Columns which are insignificant (or zero) can either be relegated to
        the end or flagged so that they don't participate *****/


        int ctr,cindex,col,n,m,startrow;
        float errold,ernew,beta,vvt,atv,wernew,werrold;

        (*sbreak)=*nc-1;

        for (ctr=0;ctr<(*nc);ctr++) s[ctr]=ctr;  //reset the permutation vector

        for (ctr=0;ctr<(*nc);ctr++)  //main loop. Find the next highest error column.

        //ctr moves through the permutation vector
        {
                errold=0;
                werrold=0;
                cindex=0;
                (ctr<*rnk) ? (startrow=ctr) : (startrow=(*rnk)-1);
                //test for rank deficiency
                // for ctr<*rnk still on diagonals

                for (n=ctr;n<(*nc);n++)
                {
                        //col=s[n];
                        ernew=0;
                        //wernew=0;
                        for (m=startrow;m<(*mr);m++)
                        {
                                if (m != startrow)
                                {
                                        //wernew += A[m][s[n]]*A[m][s[n]];
                                        ernew += A[m][s[n]]*A[m][s[n]];
                                }
                                else ernew += A[m][s[n]]*A[m][s[n]];
                        }
                        if (ernew > errold)
                                {
                                        cindex=n;
                                        //col index for the highest error so far
                                        errold=ernew;
                                        //werrold=wernew;
                                }

                }

                for (n=cindex;n>ctr;n--) SWAP(s[n],s[n-1]);
                                        //bring the highest error column to the
                                        //the next leftmost position of the permutation
                                        //vector

                if (errold < *delta)
                        //If the column error is less than the precision thenbreak out of the function
                {
                        if (ctr<*sbreak) *sbreak=ctr-1;//pass back the perm vector index where
                                                //orthogonalisation should stop
                        break;
                }

                /*** Now start the Householder decomposition ****/

                for (m=0;m<startrow;m++) {
                        v[m][s[ctr]]=0;
                }
```

```
                v[startrow][s[ctr]]=1;  // note that v is the same order as A

                beta=A[startrow][s[ctr]]+(SIGNOF(A[startrow][s[ctr]]))*(pow(errold,0.5));

                vvt=1;
                for (m=startrow+1;m<=(*mr)-1;m++)
                {
                        v[m][s[ctr]]=A[m][s[ctr]]/beta;
                        //Compute vvt
                        vvt += v[m][s[ctr]]*v[m][s[ctr]];
                }

                rbeta[s[ctr]]=-2/vvt;

                //Now update A

                for (n=ctr;n<*nc;n++)
                {
                        atv=0;
                        for (m=startrow;m<*mr;m++)
                        {
                                atv += A[m][s[n]]*v[m][s[ctr]];
                        }

                        atv=rbeta[s[ctr]]*atv;

                        for (m=startrow;m<*mr;m++)
                        {
                                A[m][s[n]] = A[m][s[n]] + atv*v[m][s[ctr]];

                        }

                }

        }
        return;
}
```

```
/**** gaussj2
            triangula matrix inversion *****/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "nrutil.h"
#define SWAP(a,b) {temp=(a);(a)=(b); (b)=temp;}

void gaussj2(float **a, int n, float **b, int m)

/******* changed 23/2/98 so that all matrices start from 0 ******/
/*To solve, for example, [A].[x1 U x2 U x3 U Y] = [b1 U b2 U b3 U 1]
so that a[0..n][..n] and b[0..n][0..m].  b width must be >= 1
a is replaced by its inverse and b by the solution vectors*/

{
        int *indxc,*indxr,*ipiv;
        int i,icol,irow,j,k,l,ll;
        float big,dum,pivinv,temp;
        indxc=ivector(1,n);
        indxr=ivector(1,n);
        ipiv=ivector(1,n);

        for (j=0;j<n; j++) ipiv[j]=0;/*main loop*/
        for (i=0;i<n;i++) {
                big=0.0;
                for (j=0;j<n;j++)
                        if (ipiv[j] != 1)
                                for (k=0;k<n;k++) {
                                        //printf("j=%d k=%d ipiv[j]=%d  ipiv[k]=%d\n",j,k,ipiv[j],ipiv[k]);
                                        if (ipiv[k] == 0) {
                                                if (fabs(a[j][k])>=big) {big=fabs(a[j][k]);irow=j;icol=k;}
                                        } else if (ipiv[k] > 1) nrerror("gaussj: Singular Matrix - 1");
                                }
                ++(ipiv[icol]);
                if (irow != icol) {for (l=0;l<n; l++) SWAP(a[irow][l],a[icol][l]);
                        for (l=0;l<m; l++) SWAP(b[irow][l],b[icol][l]);}

                indxr[i]=irow; indxc[i]=icol;
                if (a[icol][icol] == 0.0) nrerror("gaussj: Singular Matrix-2");
                pivinv=1.0/a[icol][icol];
                a[icol][icol]=1.0;
                for (l=0;l<n;l++) a[icol][l] *= pivinv;
                for (l=0;l<m;l++) b[icol][l] *= pivinv;

                for (ll=0;ll<n;ll++)
                        if (ll != icol) {
                                dum=a[ll][icol];
                                a[ll][icol]=0.0;
                                for (l=0;l<n;l++) a[ll][l] -= a[icol][l]*dum;
                                for (l=0;l<m;l++) b[ll][l] -= b[icol][l]*dum;
                        }
        }

        for (l=(n-1);l>=0;l--) {
                if (indxr[l] != indxc[l])
                        for (k=0;k<n;k++)
                                SWAP(a[k][indxr[l]],a[k][indxc[l]]);
        }


        free_ivector(ipiv,1,n);
        free_ivector(indxr,1,n);
        free_ivector(indxc,1,n);
        return;
}
```

```
/**** gaussj2
        triangula matrix inversion *****/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "nrutil.h"
#define SWAP(a,b) {temp=(a);(a)=(b); (b)=temp;}

void gaussj2(float **a, int n, float **b, int m)

/******* changed 23/2/98 so that all matrices start from 0 ******/
/*To solve, for example, [A].[x1 U x2 U x3 U Y] = [b1 U b2 U b3 U 1]
so that a[0..n][..n] and b[0..n][0..m].  b width must be >= 1
a is replaced by its inverse and b by the solution vectors*/

{
        int *indxc,*indxr,*ipiv;
        int i,icol,irow,j,k,l,ll;
        float big,dum,pivinv,temp;
        indxc=ivector(1,n);
        indxr=ivector(1,n);
        ipiv=ivector(1,n);

        for (j=0;j<n; j++) ipiv[j]=0;/*main loop*/
        for (i=0;i<n;i++) {
                big=0.0;
                for (j=0;j<n;j++)
                        if (ipiv[j] != 1)
                                for (k=0;k<n;k++) {
                                        //printf("j=%d k=%d ipiv[j]=%d  ipiv[k]=%d\n",j,k,ipiv[j],ipiv[k]);
                                        if (ipiv[k] == 0) {
                                                if (fabs(a[j][k])>=big) {big=fabs(a[j][k]);irow=j;icol=k;}
                                        } else if (ipiv[k] > 1) nrerror("gaussj: Singular Matrix - 1");
                                }
                ++(ipiv[icol]);
                if (irow != icol) {for (l=0;l<n; l++) SWAP(a[irow][l],a[icol][l]);
                                for (l=0;l<m; l++) SWAP(b[irow][l],b[icol][l]);}

                indxr[i]=irow; indxc[i]=icol;
                if (a[icol][icol] == 0.0) nrerror("gaussj: Singular Matrix-2");
                pivinv=1.0/a[icol][icol];
                a[icol][icol]=1.0;
                for (l=0;l<n;l++) a[icol][l] *= pivinv;
                for (l=0;l<m;l++) b[icol][l] *= pivinv;

                for (ll=0;ll<n;ll++)
                        if (ll != icol) {
                                dum=a[ll][icol];
                                a[ll][icol]=0.0;
                                for (l=0;l<n;l++) a[ll][l] -= a[icol][l]*dum;
                                for (l=0;l<m;l++) b[ll][l] -= b[icol][l]*dum;
                        }
        }

        for (l=(n-1);l>=0;l--) {
                if (indxr[l] != indxc[l])
                        for (k=0;k<n;k++)
                                SWAP(a[k][indxr[l]],a[k][indxc[l]]);
        }


        free_ivector(ipiv,1,n);
        free_ivector(indxr,1,n);
        free_ivector(indxc,1,n);
        return;
}
```